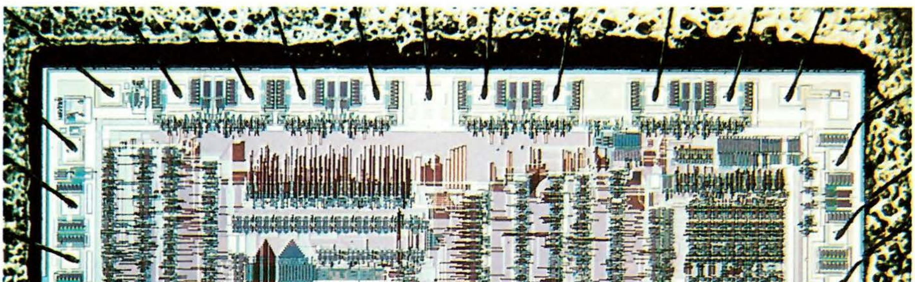
Technology and Service

Z80 CMOS FAMILY
DATAPACK



Z80C CPU

Z80 CMOS FAMILY DATAPACK

Technology and Service

Z80
THE INDUSTRY STANDARD CMOS LINE

RELIABLE CMOS P-WELL PROCESS

DIRECT SOCKET REPLACEMENT FOR STANDARD NMOS

Main Features:

- Emulator Compatible
- Very Low Power Consumption
  (15 mA Typical at 4 Mhz)
- Power Down mode Consumption less than 10 μA
- Speeds: 2.5 Mhz and 4 Mhz available
  6 Mhz coming
- Temperature Range: Ext. $-40°C/+85°C$
- Packages: Plastic, Frit-seal,
  Chip-carrier Plastic & Ceramic

LOWER CONSUMPTION FOR HIGHER RELIABILITY

Z80C CPU CMOS Version

Z84C00

Features

- The instruction set contains 158 instructions. The 78 instructions of the 8080A are included as a subset; 8080A and Z80 software compatibility is maintained.
- 4MHz and 2.5 MHz clocks for the Z80CA, and Z80C CPU result in rapid instruction execution with consequent high data throughput.
- The extensive instruction set includes string, bit, byte, and word operations. Block searches and block transfers together with indexed and relative addressing result in the most powerful data handling capabilities in the microcomputer industry.
- The Z80C microprocessors and associated family of peripheral controllers are linked by a vectored interrupt system. This system may be daisy-chained to allow implementation of a priority interrupt scheme. Little, if any, additional logic is required for daisy-chaining.
- Duplicate sets of both general-purpose and flag registers are provided, easing the design and operation of system software through single-context switching, background-foreground programming, and single-level interrupt processing. In addition, two 16-bit index registers facilitate program processing of tables and arrays.
- There are three modes of high speed interrupt processing: 8080 similar, non-Z80 peripheral device, and Z80* Family peripheral with or without daisy chain.
- On-chip dynamic memory refresh counter.
- Single 5V ± 10% Power Supply.
- Low Power Consumption:
  — 15 mA Typ. at 4 MHz
  — 9 mA Typ. at 2.5 MHz
  — Less than 10μA in Power Down mode.
- Extended Operating Temperature
  — 40°C to +85°C.

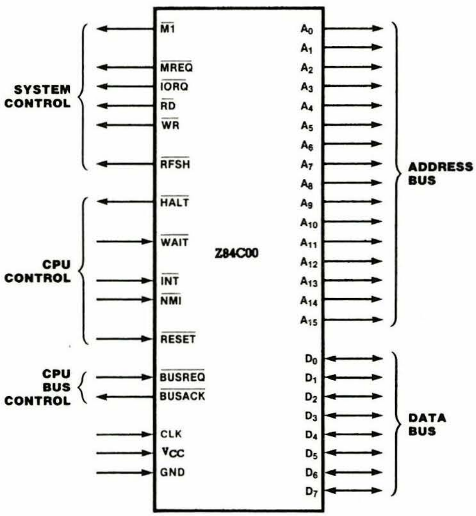


Figure 1. Logic Functions

General Description

Z80 CMOS Family is fabricated using SGS' CMOS Silicon Gate Technology, which provides low power operation and high performance.

The Z80C CPU is third-generation single-chip microprocessors with exceptional computational power. They offer higher system throughput and more efficient memory utilization than comparable second- and third-generation microprocessors. The internal registers contain 208 bits of read/write memory that are accessible to the programmer. These registers include two sets of six general-purpose registers which may be used individually as either 8-bit registers or as 16-bit register pairs. In addition, there are two sets of accumulator and flag registers. A group of "Exchange" instructions makes either set of main or alternate registers accessible to the programmer. The alternate set allows operation in foreground-background mode or it may be reserved for very fast interrupt response.

The Z80C also contains a Stack Pointer, Program Counter, two index registers, a Refresh register (counter), and an Interrupt register.

The CPU is easy to incorporate into a system since it requires only a single +5 V power source, all output signals are fully decoded and timed to control standard memory or peripheral circuits, and is supported by an extensive family of peripheral controllers.

The internal block diagram (Figure 3) shows the primary functions of the Z80C processors. Subsequent text provides more detail on the Z80C I/O controller family, registers, instruction set, interrupts and daisy chaining, and CPU timing.

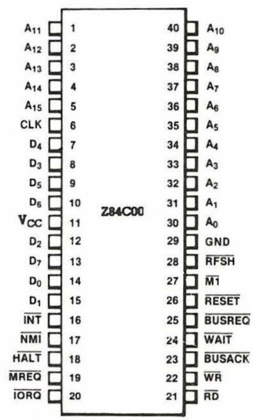


Figure 2. Pin Configuration

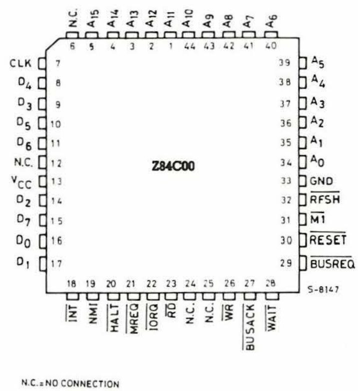


Figure 2a. Chip Carrier Pin Configuration

General Description (Continued)

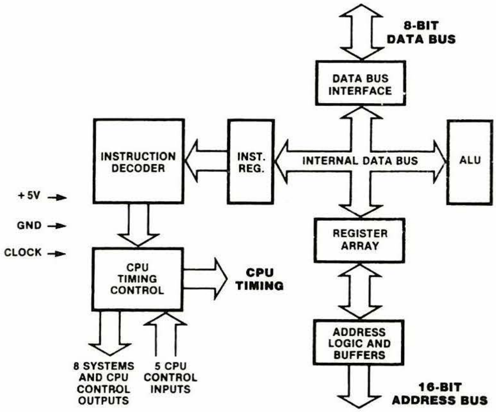


Figure 3. CPU Block Diagram

CPU Registers

Figure 4 shows three groups of registers within the CPU. The first group consists of duplicate sets of 8-bit registers: a principal set and an alternate set (designated by ' [prime],

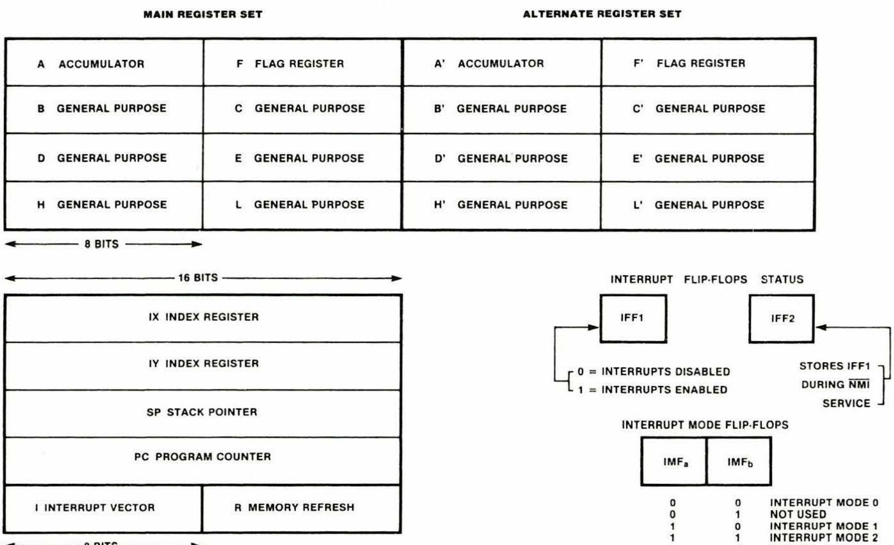


Figure 4. CPU Registers

CPU Registers (Continued)

e.g., A'). Both sets consist of the Accumulator Register, the Flag Register, and six general-purpose registers. Transfer of data between these duplicate sets of registers is accomplished by use of "Exchange" instructions. The result is faster response to interrupts and easy, efficient implementation of such versatile programming techniques as background-foreground data processing. The second set of registers consists of six registers with assigned functions. These are the I (Interrupt Register), the R (Refresh Register), the IX and IY (Index Registers), the SP (Stack Pointer), and the PC (Program Counter). The third group consists of two interrupt status flip-flops, plus and additional pair of flip-flops which assists in identifying the interrupt mode at any particular time. Table 1 provides further information on these registers.

	Register	Size (Bits)	Remarks
A, A'	Accumulator	8	Stores an operand or the results of an operation
F, F'	Flags	8	See Instruction Set.
B, B'	General Purpose	8	Can be used separately or as a 16-bit register with C
C, C'	General Purpose	8	See B, above.
D, D'	General Purpose	8	Can be used separately or as a 16-bit register with E.
E, E'	General Purpose	8	See D, above
H, H'	General Purpose	8	Can be used separately or as a 16-bit register with L.
L, L'	General Purpose	8	See H, above.
			Note: The (B,C), (D,E), and (H,L) sets are combined as follows: B – High byte C – Low byte D – High byte E – Low byte H – High byte L – Low byte
I	Interrupt Register	8	Stores upper eight bits of memory address for vectored interrupt processing.
R	Refresh Register	8	Provides user-trasparent dynamic mèmory refresh. Lower seven bits are automatically incremented and all eight are placed on the address bus during each instruction fetch cycle refresh time.
IX	Index Register	16	Used for indexed addressing.
IY	Index Register	16	Same as IX, above.
SP	Stack Pointer	16	Holds address of the top of the stack. See Push or Pop in instruction set.
PC	Program Counter	16	Holds address of next instruction.
IFF_1-IFF_2	Interrupt Enable	Flip-Flops	Set or reset to indicate interrupt status (see Figure 4).
IMFa-IMFb	Interrupt Mode	Flip-Flops	Reflect Interrupt mode (see Figure 4).

Table 1. CPU Registers

Interrupts: General Operation

The CPU accepts two interrupt input signals: $\overline{NMI}$ and $\overline{INT}$. The $\overline{NMI}$ is a non-maskable interrupt and has the highest priority. $\overline{INT}$ is a lower priority interrupt and it requires that interrupts be enabled in software in order to operate. $\overline{INT}$ can be connected to multiple peripheral devices in a wired-OR configuration.

The Z80C has a single response mode for interrupt service for the non-maskable interrupt. The maskable interrupt, $\overline{INT}$, has three programmable response modes available.
These are:

- Mode 0 — compatible with the 8080 microprocessor.
- Mode 1 — Peripheral Interrupt service, for use with non-8080/Z80 systems.
- Mode 2 — a vectored interrupt scheme, usually daisy-chained, for use with Z80 Family and compatible peripheral devices.

The CPU services interrupts by sampling the $\overline{NMI}$ and $\overline{INT}$ signals at the rising edge of the last clock of an instruction. Further interrupt service processing depends upon the type of interrupt that was detected. Details on interrupt responses are shown in the CPU Timing Section.

Non-Maskable Interrupt ($\overline{NMI}$). The non-maskable interrupt cannot be disabled by program control and therefore will be accepted at all times by the CPU. $\overline{NMI}$ is usually reserved for servicing only the highest priority type interrupts, such as that for orderly shut-down after power failure has been detected.
After recognition of the $\overline{NMI}$ signal (providing $\overline{BUSREQ}$ is not active), the CPU jumps to restart location 0066H. Normally, software starting at this address contains the interrupt service routine.

Maskable Interrupt ($\overline{INT}$). Regardless of the interrupt mode set by the user, the Z80C response to a maskable interrupt input follows a common timing cycle. After the interrupt has been detected by the CPU (provided that interrupts are enabled and $\overline{BUSREQ}$ is not active) a special interrupt processing cycle begins. This is a special fetch ($\overline{MI}$) cycle in which $\overline{IORQ}$ becomes active rather than $\overline{MREQ}$, as in normal $\overline{MI}$ cycle.
In addition, this special $\overline{MI}$ cycle is automatically extended by two $\overline{WAIT}$ states, to allow for the time required to acknowledge the interrupt request.

Mode 0 Interrupt Operation. This mode is similar with the 8080 microprocessor interrupt service procedures. The interrupting device places an instruction on the data bus.
This is normally a Restart Instruction, which will initiate a call to the selected one of eight restart locations in page zero of memory. Unlike the 8080, the Z80 CPU responds to the Call instruction with only one interrupt acknowledge cycle followed by two memory read cycles.

Mode 1 Interrupt Operation. Mode 1 operation is very similar to that for the $\overline{NMI}$. The principal difference is that the Mode 1 interrupt has a restart location of 0038H only.

Mode 2 Interrupt Operation. This interrupt mode has been designed to utilize most effectively the capabilities of the Z80C microprocessor and its associated peripheral family. The interrupting peripheral device selects the starting address of the interrupt service routine. It does this by placing an 8-bit vector on the data bus during the interrupt acknowledge cycle. The CPU forms a pointer using this byte as the lower 8-bits and the contents of the I register as the upper 8-bits. This points to an entry in a table of addresses for interrupt service routines. The CPU then jumps to the routine at that address. This flexibility in selecting the interrupt service routine address allows the peripheral device to use several different types of service routines.
These routines may be located at any available location in memory. Since the interrupting device supplies the low-order byte of the 2-byte vector, bit 0 (A_0) must be a zero.

Interrupts: General Operation (Continued)

Interrupt Priority (Daisy Chaining and Nested Interrupts). The interrupt priority of each peripheral device is determined by its physical location within a daisy-chain configuration. Each device in the chain has an interrupt enable input line (IEI) and an interrupt enable output line (IEO), which is fed to the next lower priority device. The first device in the daisy chain has its IEI input hardwired to a High level. The first device has highest priority, while each succeding device has a corresponding lower priority. This arrangement permits the CPU to select the highest priority interrupt from several simultaneously interrupting peripherals.

The interrupting device disables its IEO line to the next lower priority peripheral until it has been serviced. After servicing, its IEO line is raised, allowing lower priority peripherals to demand interrupt servicing.

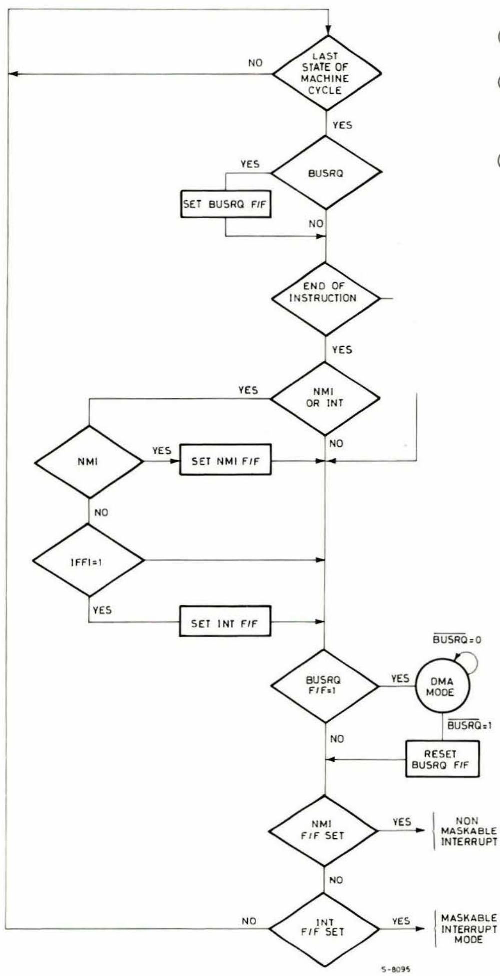
The Z80C CPU will nest (queue) any pending interrupts or interrupts received while a selected peripheral is being serviced.

Interrupt Enable/Disable Operation. Two flip-flops, IFF_1 and IFF_2, referred to in the register description are used to signal the CPU interrupt status. Operation of the two flip-flops is described in Table 2. For more details, refer to the *Z80 CPU Technical Manual.*

Action	IFF_2	IFF_2	Comments
CPU Reset	0	0	Maskable interrupt $\overline{INT}$ disabled
DI instruction execution	0	0	Maskable interrupt $\overline{INT}$ disabled
EI instruction execution	1	1	Maskable interrupt $\overline{INT}$ enabled
LD A, I instruction execution	•	•	$IFF_2 \rightarrow$ Parity flag
LD A, R instruction execution	•	•	$IFF_2 \rightarrow$ Parity flag
Accept $\overline{NMI}$	0	IFF_1	$IFF_1 \rightarrow IFF_2$ (Maskable interrupt $\overline{INT}$disabled)
RETN instruction execution	IFF_2	•	$IFF_2 \rightarrow IFF_1$ at completion of an $\overline{NMI}$ service routine.

Table 2. State of Flip-Flops

Interrupts: General Operation (Continued)



NOTE:

(1) $\overline{INT}$ and $\overline{NMI}$ are always acted on at the end of an instruction.

(2) $\overline{BUSRQ}$ is acted on at the end of a machine cycle.

(3) While the CPU is in the DMA MODE, it will not respond to active inputs on $\overline{INT}$ or $\overline{NMI}$.

(4) These three inputs are acted on in the following order of priority.

1) $\overline{BUSRQ}$ -- highest
2) $\overline{NMI}$
3) $\overline{INT}$ -- lowest

CPU Interrupt Sequence

Instruction Set

The Z80C microprocessor has one of the most powerful and versatile instruction sets available in any 8-bit microprocessor. It includes such unique operations as a block move for fast, efficient data transfers within memory or between memory and I/O. It also allows operations on any bit in any location in memory.

The following is a summary of the Z80C instruction set and shows the assembly language mnemonic, the operation, the flag status, and gives comments on each instruction. The *Z80 CPU Technical Manual and Z80 CPU Programming Manual* contain significantly more details for programming use.

The instructions are divided into the following categories:

- 8-bit loads
- 16-bit loads
- Exchanges, block transfers, and searches
- 8-bit arithmetic and logic operations
- General-purpose arithmetic and CPU control
- 16-bit arithmetic operations
- Rotates and shift
- Bit set, reset, and test operations
- Jumps
- Calls, returns, and restarts
- Input and output operations

A variety of addressing modes are implemented to permit efficient and fast data transfer between various registers, memory locations, and input/output devices. These addressing modes include:

- Immediate
- Immediate extended
- Modified page zero
- Relative
- Extended
- Indexed
- Register
- Register indirect
- Implied
- Bit

8-Bit Load Group

Mnemonic	Symbolic Operation	S	Z		H		P/V	N	C	Opcode 76 543 210	Hex	No.of Bytes	No.of M Cycles	No.of T States	Comments
LD r, r'	r ← r'	•	•	X	•	X	•	•	•	01 r r'		1	1	4	r, r' Reg.
LD r, n	r ← n	•	•	X	•	X	•	•	•	00 r 110		2	2	7	000 B
										← n →					001 C
LD r, (HL)	r ← (HL)	•	•	X	•	X	•	•	•	01 r 110		1	2	7	010 D
LD r, (IX+d)	r ← (IX+d)	•	•	X	•	X	•	•	•	11 011 101	DD	3	5	19	011 E
										01 r 101					100 H
										← d →					101 L
LD r, (IY+d)	r ← (IY+d)	•	•	X	•	X	•	•	•	11 111 101	FD	3	5	19	111 A
										01 r 110					
										← d →					
LD (HL), r	(HL) ← r	•	•	X	•	X	•	•	•	01 110 r		1	2	7	
LD (IX+d), r	(IX+d) ← r	•	•	X	•	X	•	•	•	11 011 101	DD	3	5	19	
										01 110 r					
										← d →					
LD (IY+d), r	(IY+d) ← r	•	•	X	•	X	•	•	•	11 111 101	FD	3	5	19	
										01 110 r					
										← d →					
LD (HL), n	(HL) ← n	•	•	X	•	X	•	•	•	00 110 110	36	2	3	10	
										← n →					
LD (IX+d), n	(IX+d) ← n	•	•	X	•	X	•	•	•	11 011 101	DD	4	5	19	
										00 110 110	36				
										← d →					
										← n →					
LD (IY+d), n	(IY+d) ← n	•	•	X	•	X	•	•	•	11 111 101	FD	4	5	19	
										00 110 110	36				
										← d →					
										← n →					
LD A, (BC)	A ← (BC)	•	•	X	•	X	•	•	•	00 001 010	0A	1	2	7	
LD A, (DE)	A ← (DE)	•	•	X	•	X	•	•	•	00 011 010	1A	1	2	7	
LD A, (nn)	A ← (nn)	•	•	X	•	X	•	•	•	00 111 010	3A	3	4	13	
										← n →					
										← n →					
LD (BC), A	(BC) ← A	•	•	X	•	X	•	•	•	00 000 010	02	1	2	7	
LD (DE), A	(DE) ← A	•	•	X	•	X	•	•	•	00 010 010	12	1	2	7	
LD (nn), A	(nn) ← A	•	•	X	•	X	•	•	•	00 110 010	32	3	4	13	
										← n →					
										← n →					
LD A, I	A ← I	↕	↕	X	0	X	IFF	0	•	11 101 101	ED	2	2	9	
										01 010 111	57				
LD A, R	A ← R	↕	↕	X	0	X	IFF	0	•	11 101 101	ED	2	2	9	
										01 011 111	5F				
LD I, A	I ← A	•	•	X	•	X	•	•	•	11 101 101	ED	2	2	9	
										01 000 111	47				
LD R, A	R ← A	•	•	X	•	X	•	•	•	11 101 101	ED	2	2	9	
										01 001 111	4F				

NOTES: r, r' means any of the registers A, B, C, D, E, H, L.
IFF the content of the interrupt enable flip-flop, (IFF) is copied into the P/V flag.
For an explanation of flag notation and symbols for mnemonic tables, see Symbolic Notation section following tables.

16-Bit Load Group

Mnemonic	Symbolic Operation	S	Z		H		P/V	N	C	Opcode 76 543 210	Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments
LD dd, nn	dd ← nn	•	•	X	•	X	•	•	•	00 dd0 001 ← n → ← n →		3	3	10	dd Pair 00 BC 01 DE 10 HL 11 SP
LD IX, nn	IX ← nn	•	•	X	•	X	•	•	•	11 011 101 00 100 001 ← n → ← n →	DD 21	4	4	14	
LD IY, nn	IY ← nn	•	•	X	•	X	•	•	•	11 111 101 00 100 001 ← n → ← n →	FD 21	4	4	14	
LD HL, (nn)	H ← (nn + 1) L ← (nn)	•	•	X	•	X	•	•	•	00 101 010 ← n → ← n →	2A	3	5	16	
LD dd, (nn)	dd_H ← (nn + 1) dd_L ← (nn)	•	•	X	•	X	•	•	•	11 101 101 01 dd1 011 ← n → ← n →	ED	4	6	20	
LD IX, (nn)	IX_H ← (nn + 1) IX_L ← (nn)	•	•	X	•	X	•	•	•	11 011 101 00 101 010 ← n → ← n →	DD 2A	4	6	20	
LD IY, (nn)	IY_H ← (nn + 1) IY_L ← (nn)	•	•	X	•	X	•	•	•	11 111 101 00 101 010 ← n → ← n →	FD 2A	4	6	20	
LD (nn), HL	(nn + 1) ← H (nn) ← L	•	•	X	•	X	•	•	•	00 100 010 ← n → ← n →	22	3	5	16	
LD (nn), dd	(nn + 1) ← dd_H (nn) ← dd_L	•	•	X	•	X	•	•	•	11 101 101 01 dd0 011 ← n → ← n →	ED	4	6	20	
LD (nn), IX	(nn + 1) ← IX_H (nn) ← IX_L	•	•	X	•	X	•	•	•	11 011 101 00 100 010 ← n → ← n →	DD 22	4	6	20	
LD (nn), IY	(nn + 1) ← IY_H (nn) ← IY_L	•	•	X	•	X	•	•	•	11 111 101 00 100 010 ← n → ← n →	FD 22	4	6	20	
LD SP, HL	SP ← HL	•	•	X	•	X	•	•	•	11 111 001	F9	1	1	6	
LD SP, IX	SP ← IX	•	•	X	•	X	•	•	•	11 011 101 11 111 001	DD F9	2	2	10	
LD SP, IY	SP ← IY	•	•	X	•	X	•	•	•	11 111 101 11 111 001	FD F9	2	2	10	
PUSH qq	(SP − 2) ← qq_L (SP − 1) ← qq_H SP → SP − 2	•	•	X	•	X	•	•	•	11 qq0 101		1	3	11	qq Pair 00 BC 01 DE 10 HL 11 AF
PUSH IX	(SP − 2) ← IX_L (SP − 1) ← IX_H SP → SP − 2	•	•	X	•	X	•	•	•	11 011 101 11 100 101	DD E5	2	4	15	
PUSH IY	(SP − 2) ← IY_L (SP − 1) ← IY_H SP → SP − 2	•	•	X	•	X	•	•	•	11 111 101 11 100 101	FD E5	2	4	15	
POP qq	qq_H ← (SP + 1) qq_L ← (SP) SP → SP + 2	•	•	X	•	X	•	•	•	11 qq0 001		1	3	10	
POP IX	IX_H ← (SP + 1) IX_L ← (SP) SP → SP + 2	•	•	X	•	X	•	•	•	11 011 101 11 100 001	DD E1	2	4	14	
POP IY	IY_H ← (SP + 1) IY_L ← (SP) SP → SP + 2	•	•	X	•	X	•	•	•	11 111 101 11 100 001	FD E1	2	4	14	

NOTES: dd is any of the register pairs BC, DE, HL, SP.
qq is any of the register pairs AF, BC, DE, HL.
$(PAIR)_H$, $(PAIR)_L$ refer to high order and low order eight bits of the register pair respectively,
e.g., BC_L = C, AF_H = A.

Exchange, Block Transfer, Block Search Groups

EX DE, HL	DE ↔ HL	•	•	X	•	X	•	•	•	11 101 011	EB	1	1	4	
EX AF, AF'	AF ↔ AF'	•	•	X	•	X	•	•	•	00 001 000	08	1	1	4	
EXX	BC ↔ BC' DE ↔ DE' HL ↔ HL'	•	•	X	•	X	•	•	•	11 011 001	D9	1	1	4	Register bank and auxiliary register bank exchange
EX (SP), HL	H ↔ (SP+1) L ↔ (SP)	•	•	X	•	X	•	•	•	11 100 011	E3	1	5	19	
EX (SP), IX	IX_H ↔ (SP+1) IX_L ↔ (SP)	•	•	X	•	X	•	•	•	11 011 101 11 100 011	DD E3	2	6	23	
EX (SP), IY	IY_H ↔ (SP+1) IY_L ↔ (SP)	•	•	X	•	X	•	•	•	11 111 101 11 100 011	FD E3	2	6	23	
LDI	(DE) ← (HL) DE ← DE+1 HL ← HL+1 BC ← BC−1	•	•	X	0	X	① ↕	0	•	11 101 101 10 100 000	ED A0	2	4	16	Load (HL) into (DE), increment the pointers and decrement the byte counter (BC)
LDIR	(DE) ← (HL) DE ← DE+1 HL ← HL+1 BC ← BC−1 Repeat until BC = 0	•	•	X	0	X	0	0	•	11 101 101 10 110 000	ED B0	2 2	5 4	21 16	If BC ≠ 0 If BC = 0
LDD	(DE) ← (HL) DE ← DE−1 HL ← HL−1 BC ← BC−1	•	•	X	0	X	① ↕	0	•	11 101 101 10 101 000	ED A8	2	4	16	
LDDR	(DE) ← (HL) DE ← DE−1 HL ← HL−1 BC ← BC−1 Repeat until BC = 0	•	•	X	0	X	0	0	•	11 101 101 10 111 000	ED B8	2 2	5 4	21 16	If BC ≠ 0 If BC = 0
CPI	A − (HL) HL ← HL+1 BC ← BC−1	↕	② ↕	X	↕	X	① ↕	1	•	11 101 101 10 100 001	ED A1	2	4	16	
CPIR	A − (HL) HL ← HL+1 BC ← BC−1 Repeat until A = (HL) or BC = 0	↕	② ↕	X	↕	X	① ↕	1	•	11 101 101 10 110 001	ED B1	2 2	5 4	21 16	If BC ≠ 0 and A ≠ (HL) If BC = 0 or A = (HL)
CPD	A − (HL) HL ← HL−1 BC ← BC−1	↕	② ↕	X	↕	X	① ↕	1	•	11 101 101 10 101 001	ED A9	2	4	16	
CPDR	A − (HL) HL ← HL−1 BC ← BC−1 Repeat until A = (HL) or BC = 0	↕	② ↕	X	↕	X	① ↕	1	•	11 101 101 10 111 001	ED B9	2 2	5 4	21 16	If BC ≠ 0 and A ≠ (HL) If BC = 0 or A = (HL)

NOTES: ① P/V flag is 0 if the result of BC − 1 = 0, otherwise P/V = 1.
② Z flag is 1 if A = (HL), otherwise Z = 0.

8-Bit Arithmetic and Logical Group

Mnemonic	Symbolic Operation	S	Z		H		P/V	N	C	Opcode 76 543 210	Hex	No.of Bytes	No.of M Cycles	No.of T States	Comments
ADD A, r	A ← A + r	↕	↕	X	↕	X	V	0	↕	10 $\boxed{000}$ r		1	1	4	r Reg.
ADD A, n	A ← A + n	↕	↕	X	↕	X	V	0	↕	11 $\boxed{000}$ 110		2	2	7	000 B
										← n →					001 C
															010 D
ADD A, (HL)	A ← A + (HL)	↕	↕	X	↕	X	V	0	↕	10 $\boxed{000}$ 110		1	2	7	011 E
ADD A, (IX + d)	A ← A + (IX + d)	↕	↕	X	↕	X	V	0	↕	11 011 101	DD	3	5	19	100 H
										10 $\boxed{000}$ 110					101 L
										← d →					111 A
ADD A, (IY + d)	A ← A + (IY + d)	↕	↕	X	↕	X	V	0	↕	11 111 101	FD	3	5	19	
										10 $\boxed{000}$ 110					
										← d →					
ADC A, s	A ← A + s + CY	↕	↕	X	↕	X	V	0	↕	$\boxed{001}$					s is any of r, n, (HL), (IX + d), (IY + d) as shown for ADD instruction. The indicated bits replace the $\boxed{000}$ in the ADD set above.
SUB s	A ← A − s	↕	↕	X	↕	X	V	1	↕	$\boxed{010}$					
SBC A, s	A ← A − s − CY	↕	↕	X	↕	X	V	1	↕	$\boxed{011}$					
AND s	A ← A ∧ s	↕	↕	X	1	X	P	0	0	$\boxed{100}$					
OR s	A ← A ∨ s	↕	↕	X	0	X	P	0	0	$\boxed{110}$					
XOR s	A ← A ⊕ s	↕	↕	X	0	X	P	0	0	$\boxed{101}$					
CP s	A − s	↕	↕	X	↕	X	V	1	↕	$\boxed{111}$					
INC r	r ← r + 1	↕	↕	X	↕	X	V	0	•	00 r $\boxed{100}$		1	1	4	
INC (HL)	(HL) ← (HL) + 1	↕	↕	X	↕	X	V	0	•	00 110 $\boxed{100}$		1	3	11	
INC (IX + d)	(IX + d) ← (IX + d) + 1	↕	↕	X	↕	X	V	0	•	11 011 101	DD	3	6	23	
										00 110 $\boxed{100}$					
										← d →					
INC (IY + d)	(IY + d) ← (IY + d) + 1	↕	↕	X	↕	X	V	0	•	11 111 101	FD	3	6	23	
										00 110 $\boxed{100}$					
										← d →					
DEC m	m ← m − 1	↕	↕	X	↕	X	V	1	•	$\boxed{101}$					m is any of r, (HL), (IX + d), (IY + d) as shown for INC. DEC same format and states as INC. Replace $\boxed{100}$ with $\boxed{101}$ in opcode.

General-Purpose Arithmetic and CPU Control Groups

Mnemonic	Symbolic Operation	S	Z		Flags H		P/V	N	C	Opcode 76 543 210	Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments
DAA	Converts acc. content into packed BCD following add or subtract with packed BCD operands.	↕	↕	X	↕	X	P	•	↕	00 100 111	27	1	1	4	Decimal adjust accumulator.
CPL	A ← $\overline{A}$	•	•	X	1	X	•	1	•	00 101 111	2F	1	1	4	Complement accumulator (one's complement).
NEG	A ← 0 − A	↕	↕	X	↕	X	V	1	↕	11 101 101 01 000 100	ED 44	2	2	8	Negate acc. (two's complement).
CCF	CY ← $\overline{CY}$	•	•	X	X	X	•	0	↕	00 111 111	3F	1	1	4	Complement carry flag.
SCF	CY ← 1	•	•	X	0	X	•	0	1	00 110 111	37	1	1	4	Set carry flag.
NOP	No operation	•	•	X	•	X	•	•	•	00 000 000	00	1	1	4	
HALT	CPU halted	•	•	X	•	X	•	•	•	01 110 110	76	1	1	4	
DI ★	IFF ← 0	•	•	X	•	X	•	•	•	11 110 011	F3	1	1	4	
EI ★	IFF ← 1	•	•	X	•	X	•	•	•	11 111 011	FB	1	1	4	
IM 0	Set interrupt mode 0	•	•	X	•	X	•	•	•	11 101 101 01 000 110	ED 46	2	2	8	
IM 1	Set interrupt mode 1	•	•	X	•	X	•	•	•	11 101 101 01 010 110	ED 56	2	2	8	
IM 2	Set interrupt mode 2	•	•	X	•	X	•	•	•	11 101 101 01 011 110	ED 5E	2	2	8	

NOTES: IFF indicates the interrupt enable flip-flop.
CY indicates the carry flip-flop.
★ indicates interrupts are not sampled at the end of EI or DI.

16-Bit Arithmetic Group

Mnemonic	Symbolic Operation	S	Z		H		P/V	N	C	Opcode 76 543 210	Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments
ADD HL, ss	HL ← HL + ss	•	•	X	X	X	•	0	↕	00 ss1 001		1	3	11	ss Reg. 00 BC 01 DE 10 HL 11 SP
ADC HL, ss	HL ← HL + ss + CY	↕	↕	X	X	X	V	0	↕	11 101 101 01 ss1 010	ED	2	4	15	
SBC HL, ss	HL ← HL − ss − CY	↕	↕	X	X	X	V	1	↕	11 101 101 01 ss0 010	ED	2	4	15	
ADD IX, pp	IX ← IX + pp	•	•	X	X	X	•	0	↕	11 011 101 01 pp1 001	DD	2	4	15	pp Reg. 00 BC 01 DE 10 IX 11 SP
ADD IY, rr	IY ← IY + rr	•	•	X	X	X	•	0	↕	11 111 101 00 rr1 001	FD	2	4	15	rr Reg. 00 BC 01 DE 10 IY 11 SP
INC ss	ss ← ss + 1	•	•	X	•	X	•	•	•	00 ss0 011		1	1	6	
INC IX	IX ← IX + 1	•	•	X	•	X	•	•	•	11 011 101 00 100 011	DD 23	2	2	10	
INC IY	IY ← IY + 1	•	•	X	•	X	•	•	•	11 111 101 00 100 011	FD 23	2	2	10	
DEC ss	ss ← ss − 1	•	•	X	•	X	•	•	•	00 ss1 011		1	1	6	
DEC IX	IX ← IX − 1	•	•	X	•	X	•	•	•	11 011 101 00 101 011	DD 2B	2	2	10	
DEC IY	IY ← IY − 1	•	•	X	•	X	•	•	•	11 111 101 00 101 011	FD 2B	2	2	10	

NOTES: ss is any of the register pairs BC, DE, HL, SP.
pp is any of the register pairs BC, DE, IX, SP.
rr is any of the register pairs BC, DE, IY, SP.

Rotate and Shift Group

Mnemonic	Symbolic Operation	S	Z		Flags H		P/V	N	C	Opcode 76 543 210	Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments
RLCA	CY ← 7 ← 0 ← A	•	•	X	0	X	•	0	↕	00 000 111	07	1	1	4	Rotate left circular accumulator.
RLA	CY ← 7 ← 0 A	•	•	X	0	X	•	0	↕	00 010 111	17	1	1	4	Rotate left accumulator.
RRCA	7 → 0 → CY A	•	•	X	0	X	•	0	↕	00 001 111	0F	1	1	4	Rotate right circular accumulator.
RRA	7 → 0 → CY A	•	•	X	0	X	•	0	↕	00 011 111	1F	1	1	4	Rotate right accumulator.
RLC r		↕	↕	X	0	X	P	0	↕	11 001 011 00 000 r	CB	2	2	8	Rotate left circular register r.
RLC (HL)		↕	↕	X	0	X	P	0	↕	11 001 011 00 000 110	CB	2	4	15	r Reg. 000 B 001 C 010 D 011 E 100 H 101 L 111 A
RLC (IX + d)	CY ← 7 ← 0 ← r,(HL),(IX + d),(IY + d)	↕	↕	X	0	X	P	0	↕	11 011 101 11 001 011 ← d → 00 000 110	DD CB	4	6	23	
RLC (IY + d)		↕	↕	X	0	X	P	0	↕	11 111 101 11 001 011 ← d → 00 000 110	FD CB	4	6	23	
RL m	CY ← 7 ← 0 m ≡ r,(HL),(IX + d),(IY + d)	↕	↕	X	0	X	P	0	↕	010					Instruction format and states are as shown for RLC's. To form new opcode replace 000 or RLC's with shown code.
RRC m	7 → 0 → CY m ≡ r,(HL),(IX + d),(IY + d)	↕	↕	X	0	X	P	0	↕	001					
RR m	7 → 0 → CY m ≡ r,(HL),(IX + d),(IY + d)	↕	↕	X	0	X	P	0	↕	$\boxed{011}$					
SLA m	CY ← 7 ← 0 ← 0 m ≡ r,(HL),(IX + d),(IY + d)	↕	↕	X	0	X	P	0	↕	$\boxed{100}$					
SRA m	7 → 0 → CY m ≡ r,(HL),(IX + d),(IY + d)	↕	↕	X	0	X	P	0	↕	$\boxed{101}$					
SRL m	0 → 7 → 0 → CY m ≡ r,(HL),(IX + d),(IY + d)	↕	↕	X	0	X	P	0	↕	$\boxed{111}$					
RLD	7-4 3-0 A 7-4 3-0 (HL)	↕	↕	X	0	X	P	0	•	11 101 101 01 101 111	ED 6F	2	5	18	Rotate digit left and right between the accumulator and location (HL).
RRD	7-4 3-0 A 7-4 3-0 (HL)	↕	↕	X	0	X	P	0	•	11 101 101 01 100 111	ED 67	2	5	18	The content of the upper half of the accumulator is unaffected

Bit Set, Reset and Test Group

Mnemonic	Symbolic Operation	S	Z		H		P/V	N	C	Opcode 76 543 210 Hex	No.of Bytes	No.of M Cycles	No.of T States	Comments
BIT b, r	$Z \leftarrow \overline{r_b}$	X	↕	X	1	X	X	0	•	11 001 011 CB 01 b r	2	2	8	r Reg. 000 B 001 C 010 D 011 E 100 H 101 L 111 A
BIT b, (HL)	$Z \leftarrow \overline{(HL)_b}$	X	↕	X	1	X	X	0	•	11 001 011 CB 01 b 110	2	3	12	
BIT b, $(IX+d)_b$	$Z \leftarrow \overline{(IX+d)_b}$	X	↕	X	1	X	X	0	•	11 011 101 DD 11 001 011 CB ← d → 01 b 110	4	5	20	
BIT b, $(IY+d)_b$	$Z \leftarrow \overline{(IY+d)_b}$	X	↕	X	1	X	X	0	•	11 111 101 FD 11 001 011 CB ← d → 01 b 110	4	5	20	b Bit Tested 000 0 001 1 010 2 011 3 100 4 101 5 110 6 111 7
SET b, r	$r_b \leftarrow 1$	•	•	X	•	X	•	•	•	11 001 011 CB $\boxed{11}$ b r	2	2	8	
SET b, (HL)	$(HL)_b \leftarrow 1$	•	•	X	•	X	•	•	•	11 001 011 CB $\boxed{11}$ b 110	2	4	15	
SET b, (IX+d)	$(IX+d)_b \leftarrow 1$	•	•	X	•	X	•	•	•	11 011 101 DD 11 001 011 CB ← d → $\boxed{11}$ b 110	4	6	23	
SET b, (IY+d)	$(IY+d)_b \leftarrow 1$	•	•	X	•	X	•	•	•	11 111 101 FD 11 001 011 CB ← d → $\boxed{11}$ b 110	4	6	23	
RES b, m	$m_b \leftarrow 0$ m ≡ r, (HL), (IX+d), (IY+d)	•	•	X	•	X	•	•	•	$\boxed{10}$				To form new opcode replace $\boxed{11}$ of SET b, s with $\boxed{10}$. Flags and time states for SET instruction.

NOTES: The notation m_b indicates bit b (0 to 7) or location m.

Jump Group

Mnemonic	Symbolic Operation	S	Z		H		P/V	N	C	Opcode 76 543 210 Hex	No.of Bytes	No.of M Cycles	No.of T States	Comments
JP nn	PC ← nn	•	•	X	•	X	•	•	•	11 000 011 C3 ← n → ← n →	3	3	10	
JP cc, nn	If condition cc is true PC ← nn, otherwise continue	•	•	X	•	X	•	•	•	11 cc 010 ← n → ← n →	3	3	10	cc Condition 000 NZ non-zero 001 Z zero 010 NC non-carry 011 C carry 100 PO parity odd 101 PE parity even 110 P sign positive 111 M sign negative
JR e	PC ← PC+e	•	•	X	•	X	•	•	•	00 011 000 18 ← e−2 →	2	3	12	
JR C, e	If C = 0, continue	•	•	X	•	X	•	•	•	00 111 000 38 ← e−2 →	2	2	7	If condition not met.
	If C = 1, PC ← PC+e										2	3	12	If condition is met.
JR NC, e	If C = 1, continue	•	•	X	•	X	•	•	•	00 110 000 30 ← e−2 →	2	2	7	If condition not met.
	If C = 0, PC ← PC+e										2	3	12	If condition is met.
JP Z, e	If Z = 0 continue	•	•	X	•	X	•	•	•	00 101 000 28 ← e−2 →	2	2	7	If condition not met.
	If Z = 1, PC ← PC+e										2	3	12	If condition is met.

Jump Group (Continued)

Mnemonic	Symbolic Operation	S	Z		H		P/V	N	C	Opcode 76 543 210	Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments
JR NZ, e	If Z = 1, continue	•	•	X	•	X	•	•	•	00 100 000	20	2	2	7	If condition not met.
										← e−2 →					
	If Z = 0, PC ← PC+e											2	3	12	If condition is met.
JP (HL)	PC ← HL	•	•	X	•	X	•	•	•	11 101 001	E9	1	1	4	
JP (IX)	PC ← IX	•	•	X	•	X	•	•	•	11 011 101	DD	2	2	8	
										11 101 001	E9				
JP (IY)	PC ← IY	•	•	X	•	X	•	•	•	11 111 101	FD	2	2	8	
										11 101 001	E9				
DJNZ, e	B ← B−1 If B = 0, continue	•	•	X	•	X	•	•	•	00 010 000	10	2	2	8	If B = 0.
										← e−2 →					
	If B ≠ 0, PC ← PC+e											2	3	13	If B ≠ 0.

NOTES: e represents the extension in the relative addressing mode.
e is a signed two's complement number in the range < −126, 129 >.
e−2 in the opcode provides an effective address of pc+e as PC is incremented by 2 prior to the addition of e.

Call and Return Group

Mnemonic	Symbolic Operation	S	Z		H		P/V	N	C	Opcode 76 543 210	Hex	No. of Bytes	No. of M Cycles	No. of T States	Comments
CALL nn	$(SP-1) \leftarrow PC_H$	•	•	X	•	X	•	•	•	11 001 101	CD	3	5	17	
	$(SP-2) \leftarrow PC_L$									← n →					
	$PC \leftarrow nn$									← n →					
CALL cc, nn	If condition cc is false	•	•	X	•	X	•	•	•	11 cc 100		3	3	10	If cc is false.
										← n →					
	continue, otherwise same as CALL nn									← n →		3	5	17	If cc is true.
RET	$PC_L \leftarrow (SP)$ $PC_H \leftarrow (SP+1)$	•	•	X	•	X	•	•	•	11 001 001	C9	1	3	10	
RET cc	If condition cc is false	•	•	X	•	X	•	•	•	11 cc 000		1	1	5	If cc is false.
	continue, otherwise same as RET											1	3	11	If cc is true.
RETI	Return from interrupt	•	•	X	•	X	•	•	•	11 101 101	ED	2	4	14	
										01 001 101	4D				
RETN[1]	Return from non-maskable interrupt	•	•	X	•	X	•	•	•	11 101 101	ED	2	4	14	
										01 000 101	45				
RST p	$(SP-1) \leftarrow PC_H$ $(SP-2) \leftarrow PC_L$ $PC_H \leftarrow 0$ $PC_L \leftarrow p$	•	•	X	•	X	•	•	•	11 t 111		1	3	11	

cc		Condition
000	NZ	non-zero
001	Z	zero
010	NC	non-carry
011	C	carry
100	PO	parity odd
101	PE	parity even
110	P	sign positive
111	M	sign negative

t	p
000	00H
001	08H
010	10H
011	18H
100	20H
101	28H
110	30H
111	38H

NOTE: [1]RETN loads $IFF_2 \leftarrow IFF_1$

Input and Output Group

Mnemonic	Symbolic Operation	S	Z		H		P/V	N	C	Opcode 76 543 210	Hex	No.of Bytes	No.of M Cycles	No.of T States	Comments
IN A, (n)	A ← (n)	•	•	X	•	X	•	•	•	11 011 011 ← n →	DB	2	3	11	n to A_0 ~ A_7 Acc. to A_8 ~ A_{15}
IN r, (C)	r ← (C) if r = 110 only the flags will be affected	↕	↕	X	↕	X	P	0	•	11 101 101 01 r 000	ED	2	3	12	C to A_0 ~ A_7 B to A_8 ~ A_{15}
INI	(HL) ← (C) B ← B − 1 HL ← HL + 1	X	① ↕	X	X	X	X	1	•	11 101 101 10 100 010	ED A2	2	4	16	C to A_0 ~ A_7 B to A_8 ~ A_{15}
INIR	(HL) ← (C) B ← B − 1 HL ← HL + 1 Repeat until B = 0	X	1	X	X	X	X	1	•	11 101 101 10 110 010	ED B2	2 2	5 (If B≠0) 4 (If B = 0)	21 16	C to A_0 ~ A_7 B to A_8 ~ A_{15}
IND	(HL) ← (C) B ← B − 1 HL ← HL − 1	X	① ↕	X	X	X	X	1	•	11 101 101 10 101 010	ED AA	2	4	16	C to A_0 ~ A_7 B to A_8 ~ A_{15}
INDR	(HL) ← (C) B ← B − 1 HL ← HL − 1 Repeat until B = 0	X	1	X	X	X	X	1	•	11 101 101 10 111 010	ED BA	2 2	5 (If B≠0) 4 (If B = 0)	21 16	C to A_0 ~ A_7 B to A_8 ~ A_{15}
OUT (n), A	(n) ← A	•	•	X	•	X	•	•	•	11 010 011 ← n →	D3	2	3	11	n to A_0 ~ A_7 Acc. to A_8 ~ A_{15}
OUT (C), r	(C) ← r	•	•	X	•	X	•	•	•	11 101 101 01 r 001	ED	2	3	12	C to A_0 ~ A_7 B to A_8 ~ A_{15}
OUTI	(C) ← (HL) B ← B − 1 HL ← HL + 1	X	① ↕	X	X	X	X	1	•	11 101 101 10 100 011	ED A3	2	4	16	C to A_0 ~ A_7 B to A_8 ~ A_{15}
OTIR	(C) ← (HL) B ← B − 1 HL ← HL + 1 Repeat until B = 0	X	1	X	X	X	X	1	•	11 101 101 10 110 011	ED B3	2 2	5 (If B≠0) 4 (If B = 0)	21 16	C to A_0 ~ A_7 B to A_8 ~ A_{15}
OUTD	(C) ← (HL) B ← B − 1 HL ← HL − 1	X	① ↕	X	X	X	X	1	•	11 101 101 10 101 011	ED AB	2	4	16	C to A_0 ~ A_7 B to A_8 ~ A_{15}
OTDR	(C) ← (HL) B ← B − 1 HL ← HL − 1 Repeat until B = 0	X	1	X	X	X	X	1	•	11 101 101 10 111 011	ED	2 2	5 (If B≠0) 4 (If B = 0)	21 16	C to A_0 ~ A_7 B to A_8 ~ A_{15}

NOTE: ① If the result of B − 1 is zero the Z flag is set, otherwise it is reset.

Summary of Flag Operation

Symbol	Operation
S	Sign flag. S = 1 if the MSB of the result is 1.
Z	Zero flag. Z = 1 if the result of the operation is 0.
P/V	Parity or overflow flag. Parity (P) and overflow (V) share the same flag. Logical operations affect this flag with the parity of the result while arithmetic operations affect this flag with the overflow of the result. If P/V holds parity, P/V = 1 if the result of the operation is even, P/V = 0 if result is odd. If P/V holds overflow, P/V = 1 if the result of the operation produced an overflow..
H	Half-carry flag. H = 1 if the add or subtract operation produced a carry into or borrow from bit 4 of the accumulator.
N	Add/Subtract flag. N = 1 if the previous operation was a subtract.
H & N	H and N flags are used in conjunction with the decimal adjust instruction (DAA) to properly correct the result into packed BCD format following addition or subtraction using operands with packed BCD format.
C	Carry/Link flag. C = 1 if the operation produced a carry from the MSB of the operand or result.

Symbol	Operation
↕	The flag is affected according to the result of the operation.
•	The flag is unchanged by the operation.
0	The flag is reset by the operation.
1	The flag is set by the operation.
X	The flag is a "don't care."
V	P/V flag affected according to the overflow result of the operation.
P	P/V flag affected according to the parity result of the operation.
r	Any one of the CPU registers A, B, C, D, E, H, L.
s	Any 8-bit location for all the addressing modes allowed for the particular instruction.
ss	Any 16-bit location for all the addressing modes allowed for that instruction.
ii	Any one of the two index registers IX or IY.
R	Refresh counter.
n	8-bit value in range < 0, 255 >.
nn	16-bit value in range < 0, 65535 >.

Symbolic Notation

Instruction	D_7 S	Z		H		P/V	N	D_0 C	Comments
ADD A, s; ADC A, s	↕	↕	X	↕	X	V	0	↕	8-bit add or add with carry.
SUB s; SBC A, s, CP s; NEG	↕	↕	X	↕	X	V	1	↕	8-bit subtract, subtract with carry, compare and negate accumulator.
AND s	↕	↕	X	1	X	P	0	0	Logical operations.
OR s, XOR s	↕	↕	X	0	X	P	0	0	
INC s	↕	↕	X	↕	X	V	0	•	8-bit increment.
DEC s	↕	↕	X	↕	X	V	1	•	8-bit decrement.
ADD DD, ss	•	•	X	X	X	•	0	↕	16-bit add.
ADC HL, ss	↕	↕	X	X	X	V	0	↕	16-bit add with carry.
SBC HL, ss	↕	↕	X	X	X	V	1	↕	16-bit subtract with carry.
RLA, RLCA, RRA; RRCA	•	•	X	0	X	•	0	↕	Rotate accumulator.
RL m; RLC m; RR m; RRC m; SLA m; SRA m; SRL m	↕	↕	X	0	X	P	0	↕	Rotate and shift locations.
RLD; RRD	↕	↕	X	0	X	P	0	•	Rotate digit left and right.
DAA	↕	↕	X	↕	X	P	•	↕	Decimal adjust accumulator.
CPL	•	•	X	1	X	•	1	•	Complement accumulator.
SCF	•	•	X	0	X	•	0	1	Set carry.
CCF	•	•	X	X	X	•	0	↕	Complement carry.
IN r (C)	↕	↕	X	0	X	P	0	•	Input register indirect.
INI, IND, OUTI; OUTD	X	↕	X	X	X	X	1	•	Block input and output. Z = 0 if B ≠ 0 otherwise Z = 0.
INIR; INDR; OTIR; OTDR	X	1	X	X	X	X	1	•	
LDI; LDD	X	X	X	0	X	↕	0	•	Block transfer instructions. P/V = 1 if BC ≠ 0, otherwise P/V = 0.
LDIR; LDDR	X	X	X	0	X	0	0	•	
CPI; CPIR; CPD; CPDR	X	↕	X	X	X	↕	1	•	Block search instructions. Z = 1 if A = (HL), otherwise Z = 0. P/V = 1 if BC ≠ 0, otherwise P/V = 0.
LD A, I, LD A, R	↕	↕	X	0	X	IFF	0	•	The content of the interrupt enable flip-flop (IFF) is copied into the P/V flag.
BIT b, s	X	↕	X	1	X	X	0	•	The state of bit b of location s is copied into the Z flag.

Pin Descriptions

A_0-A_{15}. *Address Bus* (output, active High, 3-state). A_0-A_{15} form a 16-bit address bus. The Address Bus provides the address for memory data bus exchanges (up to 64K bytes) and for I/O device exchanges.

BUSACK. *Bus Acknowledge* (output, active Low). Bus Acknowledge indicates to the requesting device that the CPU address bus, data bus, and control signals $\overline{\text{MREQ}}$, $\overline{\text{IORQ}}$, $\overline{\text{RD}}$, and $\overline{\text{WR}}$ have entered their high-impedance states. The external circuitry can now control these lines.

BUSREQ. *Bus Request* (input, active Low). Bus Request has a higher priority than $\overline{\text{NMI}}$ and is always recognized at the end of the current machine cycle. $\overline{\text{BUSREQ}}$ forces the CPU address bus, data bus, and control signals $\overline{\text{MREQ}}$, $\overline{\text{IORQ}}$, $\overline{\text{RD}}$, and $\overline{\text{WR}}$ to go to a high-impedance state so that other devices can control these lines. $\overline{\text{BUSREQ}}$ is normally wire-ORed and requires an external pullup for these applications. Extended $\overline{\text{BUSREQ}}$ periods due to extensive DMA operations can prevent the CPU from properly refreshing dynamic RAMs.

D_0-D_7. *Data Bus* (input/output), active High, 3-state). D_0-D_7 constitute an 8-bit bidirectional data bus, used for data exchanges with memory and I/O.

HALT. *Halt State* (output, active Low). $\overline{\text{HALT}}$ indicates that the CPU has executed a Halt instruction and is awaiting either a non-maskable or a maskable interrupt (with the mask enabled) before operation can be resumed.
While halted, the CPU executes NOPs to maintain memory refresh.

INT. *Interrupt Request* (input, active Low). Interrupt Request is generated by I/O devices. The CPU honors a request at the end of the current instruction if the internal software-controlled interrupt enable flip-flop (IFF) is enabled. $\overline{\text{INT}}$ is normally wire-ORed and requires an external pullup for these applications.

IORQ. *Input/Output Request* (output, active Low, 3-state). $\overline{\text{IORQ}}$ indicates that the lower half of the address bus holds a valid I/O address for an I/O read or write operation. $\overline{\text{IORQ}}$ is also generated concurrently with $\overline{\text{M1}}$ during an interrupt acknowledge cycle to indicate that an interrupt response vector can be placed on the data bus.

M1. *Machine Cycle One* (output, active Low). $\overline{\text{M1}}$, together with $\overline{\text{MREQ}}$, indicates that the current machine cycle is the opcode fetch cycle of an instruction execution. $\overline{\text{M1}}$, together with $\overline{\text{IORQ}}$, indicates an interrupt acknowledge cycle.

MREQ. *Memory Request* (output, active Low, 3-state). $\overline{\text{MREQ}}$ indicates that the address bus holds a valid address for a memory read or memory write operation.

NMI. *Non-Maskable Interrupt* (input, negative edge-triggered). $\overline{\text{NMI}}$ has a higher priority than $\overline{\text{INT}}$. $\overline{\text{NMI}}$ is always recognized at the end of the current instruction, independent of the status of the interrupt enable flip-flop, and automatically forces the CPU to restart at location 0066H.

RD. *Read* (output, active Low, 3-state). $\overline{\text{RD}}$ indicates that the CPU wants to read data from memory or an I/O device. The addressed I/O device or memory should use this signal to gate data onto the CPU data bus.

RESET. *Reset* (input, active Low). $\overline{\text{RESET}}$ initializes the CPU as follows: it resets the interrupt enable flip-flop, clears the PC and Registers I and R, and sets the interrupt status to Mode 0. During reset time, the address and data bus go to a high-impedance state, and all control output signals go to the inactive state.
Note that $\overline{\text{RESET}}$ must be active for a minimum of three full clock cycles before the reset operation is complete.

RFSH. *Refresh* (output, active Low). $\overline{\text{RFSH}}$, together with $\overline{\text{MREQ}}$, indicates that the lower seven bits of the system's address bus can be used as a refresh address to the system's dynamic memories.

WAIT. *Wait* (input, active Low). $\overline{\text{WAIT}}$ indicates to the CPU that the addressed memory or I/O devices are not ready for a data transfer. The CPU continues to enter a Wait state as long as this signal is active. Extended $\overline{\text{WAIT}}$ periods can prevent the CPU from refreshing dynamic memory properly.

WR. *Write* (output, active Low, (3-state). $\overline{\text{WR}}$ indicates that the CPU data bus holds valid data to be stored at the addressed memory or I/O location.

CPU Timing

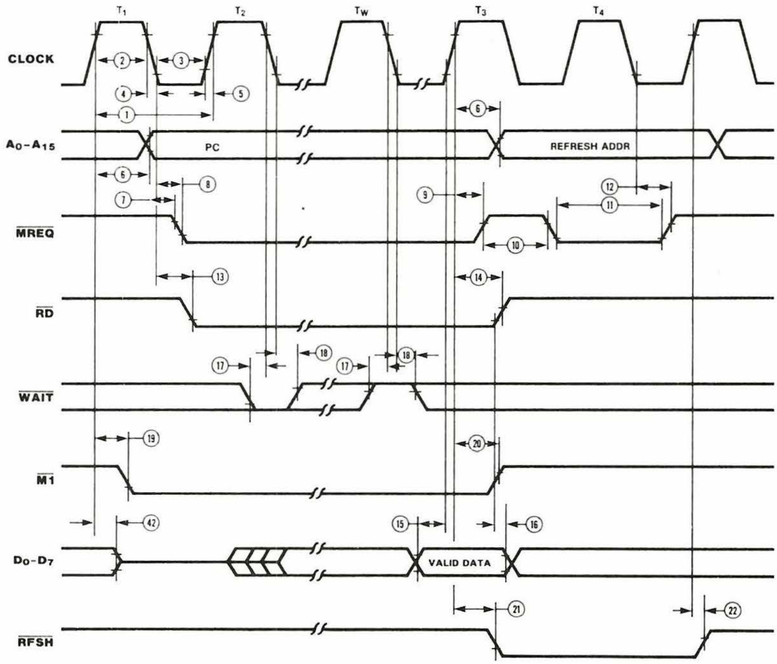
The Z80C CPU executes instructions by proceeding through a specific sequence of operations:

- Memory read or write
- I/O device read or write
- Interrupt acknowledge

The basic clock period is referred to as a T time or cycle, and three or more T cycles make up a machine cycle (M1, M2 or M3 for instance). Machine cycles can be extended either by the CPU automatically inserting one or more Wait states or by the insertion of one or more Wait states by the user.

Instruction Opcode Fetch. The CPU places the contents of the Program Counter (PC) on the address bus at the start of the cycle (Figure 5). Approximately one-half clock cycle later, $\overline{MREQ}$ goes active. When active, $\overline{RD}$ indicates that the memory data can be enabled onto the CPU data bus.

The CPU samples the $\overline{WAIT}$ input with the falling edge of clock state T_2. During clock states T_3 and T_4 of an $\overline{M1}$ cycle dynamic RAM refresh can occur while the CPU starts decoding and executing the instruction. When the Refresh Control signal becomes active, refreshing of dynamic memory can take place.



NOTE: T_W-Wait cycle added when necessary for slow ancilliary devices.

Figure 5. Instruction Opcode Fetch

CPU Timing (Continued)

Memory Read or Write Cycles. Figure 6 shows the timing of memory read or write cycles other than an opcode fetch ($\overline{M1}$) cycle. The $\overline{MREQ}$ and $\overline{RD}$ signals function exactly as in the fetch cycle. In a memory write cycle, $\overline{MREQ}$ also becomes active when the address bus is stable. The $\overline{WR}$ line is active when the data bus is stable, so that it can be used directly as an R/$\overline{W}$ pulse to most semiconductor memories.

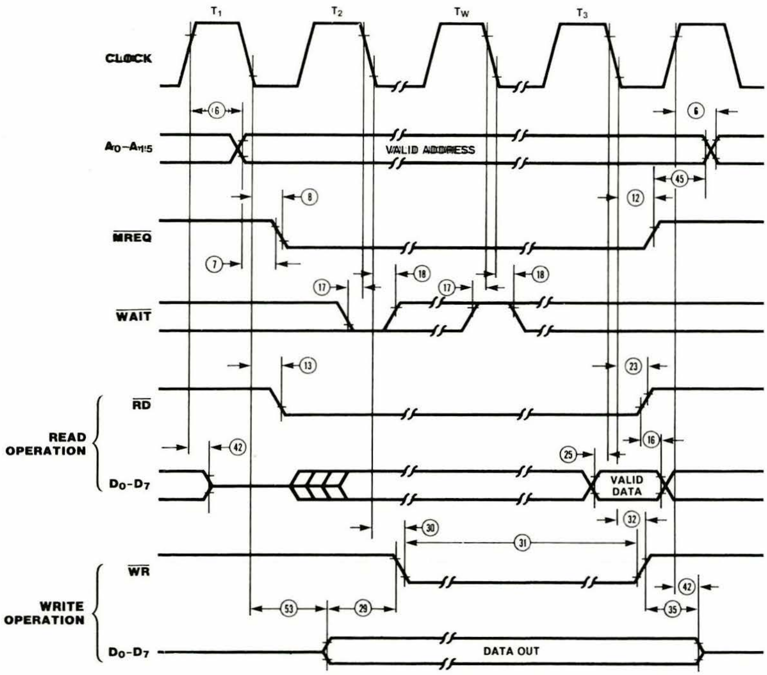
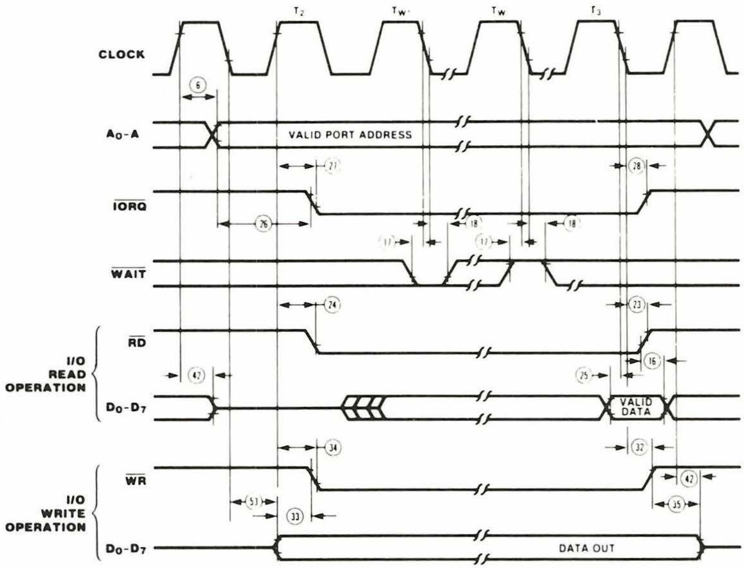


Figure 6. Memory Read or Write Cycles

CPU Timing (Continued)

Input or Output Cycles. Figure 7 shows the timing for an I/O read or I/O write operation. During I/O operations, the CPU automatically inserts a single Wait state (T_W). This extra Wait state allows sufficient time for an I/O port to decode the address from the port address lines.



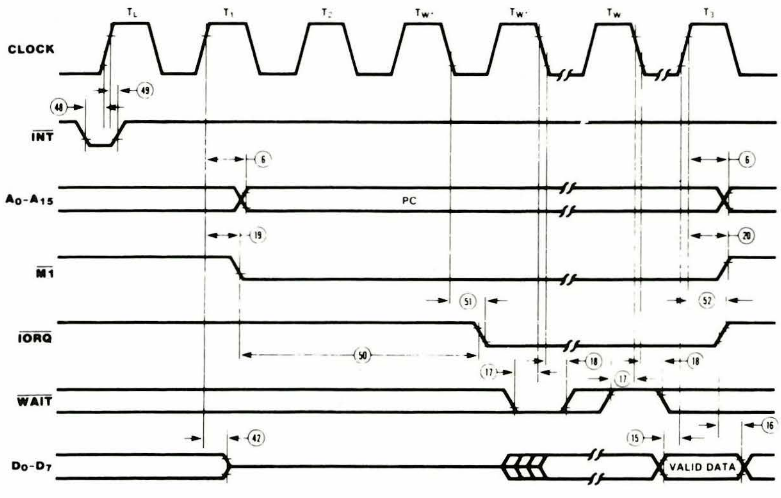
NOTE: T_W* = One Wait cycle automatically inserted by CPU.

Figure 7. Input or Output Cycles

CPU Timing (Continued)

Interrupt Request/Acknowledge Cycle. The CPU samples the interrupt signal with the rising edge of the last clock cycle at the end of any instruction (Figure 8). When an interrupt is accepted, a special $\overline{M1}$ cycle is generated.

During this $\overline{M1}$ cycle, $\overline{IORQ}$ becomes active (instead of $\overline{MREQ}$) to indicate that the interrupting device can place an 8-bit vector on the data bus. The CPU automatically adds two Wait states to this cycle.

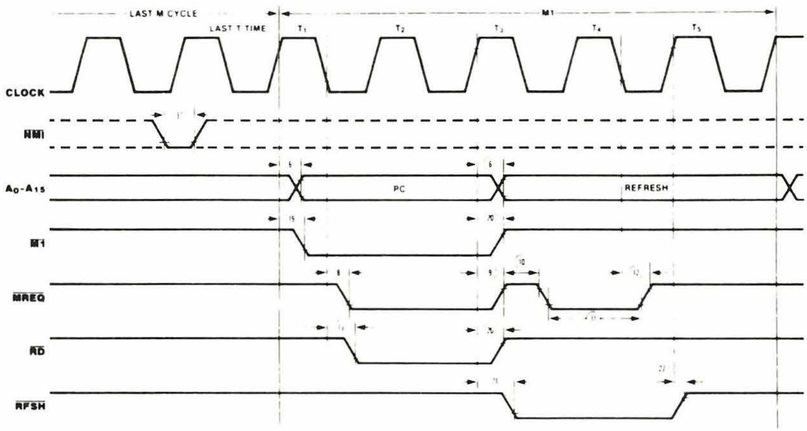


NOTE: 1) T_L = Last state of previous instruction. 2) Two Wait cycles automatically inserted by CPU(*).

Figure 8. Interrupt Request/Acknowledge Cycle

CPU Timing (Continued)

Non-Maskable Interrupt Request Cycle. $\overline{NMI}$ is sampled at the same time as the maskable interrupt input $\overline{INT}$ but has higher priority and cannot be disabled under software control. The subsequent timing is similar to that of a normal instruction fetch except that data put on the bus by the memory is ignored. The CPU instead executes a restart (RST) operation and jumps to the $\overline{NMI}$ service routine located at address 0066H (Figure 9).

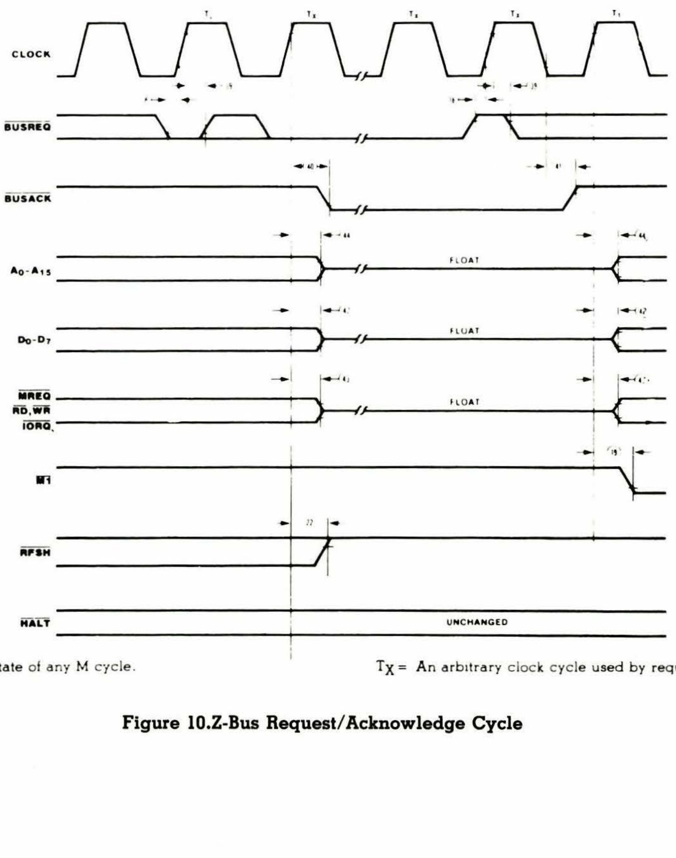


*Although $\overline{NMI}$ is an asynchronous input, to guarantee its being recognized on the following machine cycle, $\overline{NMI}$'s falling edge must occur no later than the rising edge of the clock cycle preceding T_{LAST}.

Figure 9. Non-Maskable Interrupt Request Operation

CPU Timing (Continued)

Bus Reqeust/Acknowledge Cycle. The CPU samples $\overline{BUSREQ}$ with the rising edge of the last clock period of any machine cycle (Figure 10). If $\overline{BUSREQ}$ is active, the CPU sets its address, data, and $\overline{MREQ}$, $\overline{IORQ}$, $\overline{RD}$, and $\overline{WR}$ lines to a high-impedance state with the rising edge of the next clock pulse. At that time, any external device can take control of these lines, usually to transfer data between memory and I/O devices.



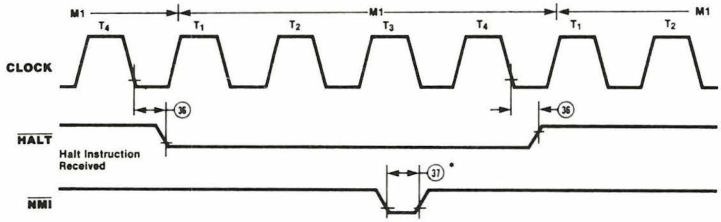
NOTE: T_L = Last state of any M cycle. T_X = An arbitrary clock cycle used by requesting device.

Figure 10.Z-Bus Request/Acknowledge Cycle

CPU Timing (Continued)

Halt Acknowledge Cycle. When the CPU receives an Halt instruction, it executes NOP states until either an $\overline{INT}$ or $\overline{NMI}$ input is received. When in the Halt state, the $\overline{HALT}$ output is active and remains so until an interrupt is processed (Figure 11).

Reset Cycle. $\overline{RESET}$ must be active for at least three clock cycles for the CPU to properly accept it. As long as $\overline{RESET}$ remains active, the address and data buses float, and the control outputs are inactive. Once $\overline{RESET}$ goes inactive, three internal T cycles are consumed before the CPU resumes normal processing operation. $\overline{RESET}$ clears the PC register, so the first opcode fetch will be to location 0000 (Figure 12).



NOTE: $\overline{INT}$ will also force a Halt exit.

*See note, Figure 9.

Figure 11. Halt Acknowledge Cycle

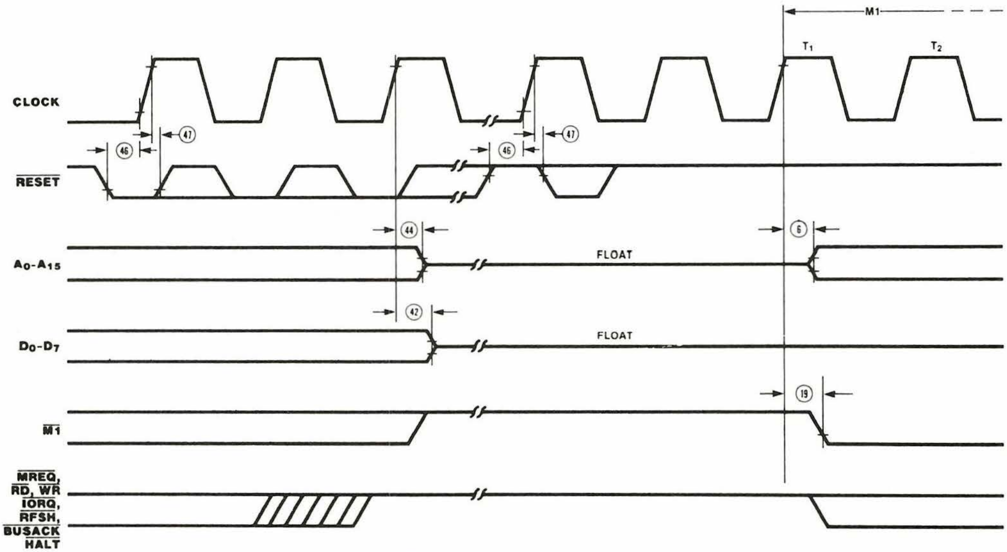


Figure 12. Reset Cycle

CPU Timing (Continued)

Power Down. When the CPU system clock is stopped at either a high or low level, the CPU stops its operation and maintains registers and control signals.

However I_{CC2} Stand-by Supply Current is guaranteed only when the supplied system clock is stopped at a low level during T4 state of the following machine cycle (actually that is M1 cycle and executes NOP instruction) next to OPcode fetch cycle of HALT instruction. The timing diagram when POWER DOWN function is implemented by HALT instruction is shown in Figure 13.

This function can be easily realized when a clock generator controller is connected with the CPU.

Release From Power Down State. The system clock must be supplied to the CPU to release power down state.
When the system clock is supplied to the CPU CLK terminal, CPU restarts operation continuously from the state when power down function has been implemented.

Note the followings when release from power down state.

(1) When external oscillator has been stopped to enter power down state, some warming-up time may be required to obtain precious and stable system clock for release from power down state.

(2) When HALT instruction is executed to enter power down state, the CPU will enter HALT state. An interrupt signal ($\overline{NMI}$ or $\overline{INT}$) or $\overline{RESET}$ signal must be generated after the system clock is supplied to release power down state. Otherwise the CPU is still in HALT state even if the system clock is supplied.

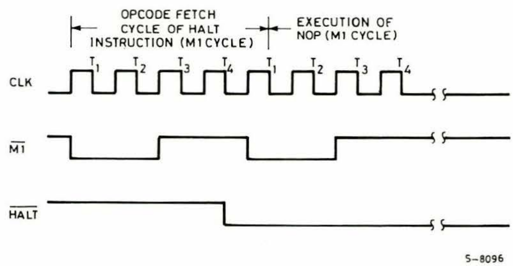


Figure 13. Timing Diagram of Power Down Function by Halt Instruction

AC Characteristics

Number	Symbol	Parameter	Z84C00 Min (ns)	Z84C00 Max (ns)	Z84C00A Min (ns)	Z84C00A Max (ns)
1	TcC	Clock Cycle Time	400	DC	250	DC
2	TwCh	Clock Pulse Width (High)	180	DC	110	DC
3	TwCl	Clock Pulse Width (Low)	180	DC	110	DC
4	TfC	Clock Fall Time	—	30	—	30
5	TrC	Clock Rise Time	—	30	—	30
6	TdCr(A)	Clock ↑ to Address Valid Delay	—	145	—	110
7	TdA(MREQf)	Address Valid to $\overline{MREQ}$ ↓ Delay	125	—	65	—
8	TdCf(MREQf)	Clock ↓to $\overline{MREQ}$ ↓ Delay	—	100	—	85
9	TdCr(MREQr)	Clock ↑ to $\overline{MREQ}$ ↑ Delay	—	100	—	85
10	TwMREQh	$\overline{MREQ}$ Pulse Width (High)	170	—	110	—
11	TwMREQl	$\overline{MREQ}$ Pulse Width (Low)	360	—	220	—
12	TdCf(MREQr)	Clock ↓ to $\overline{MREQ}$ ↑ Delay	—	100	—	85
13	TdCf(RDf)	Clock ↓ to $\overline{RD}$ ↓ Delay	—	130	—	95
14	TdCr(RDr)	Clock ↑ to $\overline{RD}$ ↑ Delay	—	100	—	85
15	TsD(Cr)	Data Setup Time to Clock ↑	50	—	35	—
16	ThD(RDr)	Data Hold Time to $\overline{RD}$ ↑	0	—	0	—
17	TsWAIT(Cf)	$\overline{WAIT}$ Setup Time to Clock ↓	70	—	70	—
18	ThWAIT(Cf)	$\overline{WAIT}$ Hold Time after Clock ↓	20	—	10	—
19	TdCr(M1f)	Clock ↑ to $\overline{M1}$ ↓ Delay	—	130	—	100
20	TdCr(M1r)	Clock ↑ to $\overline{M1}$ ↑ Delay	—	130	—	100
21	TdCr(RFSHf)	Clock ↑ to $\overline{RFSH}$ ↓ Dealy	—	180	—	130
22	TdCr(RFSHr)	Clock ↑ to $\overline{RFSH}$ ↑ Delay	—	150	—	120
23	TdCf(RDr)	Clock ↓ to $\overline{RD}$ ↑ Delay	—	110	—	85
24	TdCr(RDf)	Clock ↑ to $\overline{RD}$ ↓ Dealy	—	110	—	85
25	TsD(Cf)	Data Setup to Clock ↓ during M_2, M_3, M_4 or M_5 Cycles	60	—	50	—
26	TdA(IORQf)	Address Stable prior to $\overline{IORQ}$ ↓	320	—	180	—
27	TdCr(IORQf)	Clock ↑ to $\overline{IORQ}$ ↓ Delay	—	100	—	75
28	TdCf(IORQr)	Clock ↓ to $\overline{IORQ}$ ↑ Delay	—	110	—	85
29	TdCf(WRf)	Data Stable prior to $\overline{WR}$ ↓	190	—	80	—
30	TdDf(WRf)	Clock ↓ to $\overline{WR}$ ↓ Delay	—	90	—	80
31	TwWR	$\overline{WR}$ Pulse Width	360	—	220	—
32	TdCf(WRr)	Clock ↓ to $\overline{WR}$ ↑ Delay	—	100	—	80
33	TdD(WRf)	Data Stable prior to $\overline{WR}$ ↓	20	—	10	—
34	TdCr(WRf)	Clock ↑ to $\overline{WR}$ ↓ Delay	—	100	—	65
35	TdWRr(D)	Data Stable from $\overline{WR}$ ↑	120	—	60	—
36	TdCf(HALT)	Clock ↓ to $\overline{HALT}$ ↑ or ↓	—	300	—	300
37	TwNMI	$\overline{NMI}$ Pulse Width	80	—	80	—
38	TsBUSREQ(Cr)	$\overline{BUSREQ}$ Setup Time to Clock ↑	80	—	50	—

AC Characteristics (Continued)

Number	Symbol	Parameter	Z84C00 Min (ns)	Z84C00 Max (ns)	Z84C00A Min (ns)	Z84C00A Max (ns)
39	TcBUSUREQ(Cr)	$\overline{\text{BUSREQ}}$ Hold Time after Clock ↑	20	—	10	—
40	TdCr(BUSACKf)	Clock ↑ to $\overline{\text{BUSACK}}$ ↓ Delay	—	120	—	100
41	TdCf(BUSACKr)	Clock ↓ to $\overline{\text{BUSACK}}$ ↑ Delay	—	110	—	100
42	TdCr(Tz)	Clock ↑ to Data Float Delay	—	90	—	90
43	TdCr(CTz)	Clock ↑ to Control Outputs Float Delay ($\overline{\text{MREQ}}$, $\overline{\text{IORQ}}$, $\overline{\text{RD}}$, and $\overline{\text{WR}}$)	—	110	—	80
44	TdCr(Az)	Clock ↑ to Address Float Delay	—	110	—	90
45	TdCTr(A)	$\overline{\text{MREQ}}$ ↑, $\overline{\text{IORQ}}$ ↑, $\overline{\text{RD}}$ ↑, and $\overline{\text{WR}}$ ↑ to Address Hold Time	160	—	80	—
46	TsRESET(Cr)	$\overline{\text{RESET}}$ to Clock ↑ Setup Time	90	—	60	—
47	ThRESET(Cr)	$\overline{\text{RESET}}$ to Clock ↑ Hold Time	20	—	10	—
48	TsINTf(Cr)	$\overline{\text{INT}}$ to Clock ↑ Setup Time	80	—	80	—
49	ThINTr(Cr)	$\overline{\text{INT}}$ to Clock ↑ Hold Time	20	—	10	—
50	TdMIf(IORQf)	$\overline{\text{MI}}$ ↓ to $\overline{\text{IORQ}}$ ↓ Delay	920	—	565	—
51	TdCf(IORQf)	Clock ↓ to $\overline{\text{IORQ}}$ ↓ Delay	—	110	—	85
52	TdCf(IORQr)	Clock ↑ to $\overline{\text{IORQ}}$ ↑ Delay	—	115	—	85
53	TdCf(D)	Clock ↓ to Data Valid Delay	—	230	—	150

Absolute Maximum Ratings

Simbol	Item	Rating
V_{CC}	V_{CC} Supply Voltage with respect to V_{SS}	-0.5 to 7V
V_{IN}	Input Voltage	-0.5V to $V_{CC}+0.5$V
P_D	Power Dissipation ($T_A = 85°C$)	250mW
T_{SOLDER}	Soldering Temperature (Soldering Time 10 sec)	260°C
T_{STG}	Storage Temperature	-65°C to 150°C
T_{OPR}	Operating Temperature	-40°C to 85°C

DC Characteristics

Symbol	Parameter	Min.	Typ.	Max	Unit	Test Conditions
V_{ILC}	Clock Input Low Voltage	-0.3	—	0.6	V	
V_{IHC}	Clock Input High Voltage	$V_{CC}-0.6$	—	$V_{CC}+0.3$	V	
V_{IL}	Input Low Voltage (except CLK)	-0.5	—	0.8	V	
V_{IH}	Input High Voltage (except CLK)	2.2	—	V_{CC}	V	
V_{OL}	Output Low Voltage	—	—	0.4	V	$I_{OL}=2.0$mA
V_{OH1}	Output High Voltage (1)	2.4	—	—	V	$I_{OH}=-1.6$mA
V_{OH2}	Output High Voltage (2)	$V_{CC}-0.8$	—	—	V	$I_{OH}=-250\mu$A
I_{LI}	Input Leakage Current	—	—	±10	μA	$V_{SS} \leq V_{IN} \leq V_{CC}$
I_{LO}	3-State Output Leakage Current in Float	—	—	10	μA	$V_{SS}+0.4 \leq V_{OUT} \leq V_{CC}$
I_{CC1}	Operating Supply Current	—	15	25	mA	$V_{CC}=5$V, CLK = 4MHz $V_{IH}=V_{CC}-0.2$V $V_{IL}=0.2$V
$I_{CC2(1)}$	Stand-by Supply Current	—	0.5	10	μA	$V_{CC}=5$V CLK = (1) $V_{IL}=V_{CC}-0.2$V $V_{IH}=0.2$V

Note (1): I_{CC2} Stand-by Supply Current is guaranteed only when the supplied clock is stopped at a low level during T4 state of the following machine cycle (M1) next to OPcode fetch cycle of HALT instruction.

Test Conditions

$T_A = -40°C$ to $+85°C$
$V_{CC} = 5V \pm 10\%$
$V_{SS} = 0V$

AC test conditions

- Inputs except CLK (clock) are driven at 2.4V for a logic "1" and 0.4V for a logic "0". Clock input is driven at $V_{CC} - 0.6V$ for a logic "1" and 0.6V for a logic "0".
- Timing measurements are made at 2.2V for a logic "1" and 0.8V for a logic "0".

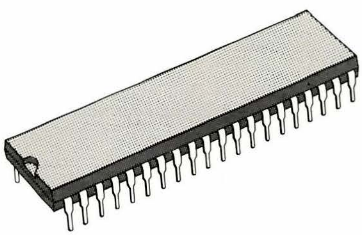
All AC parameters assume a load capacitance of 100pF

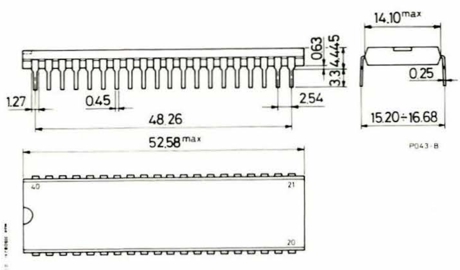
Ordering Information

Type	Package	Temp.	Clock	Description
Z84C00 B6	Plastic	−40/+85°C	2.5 MHz	Z80C Central Processing Unit CMOS Version
Z84C00 F6	Frit Seal	−40/+85°C		
Z84C00 C6	Plastic Chip-Carrier	−40/+85°C		
Z84C00 K6	Ceramic Chip-Carrier	−40/+85°C		
Z84C00A B6	Plastic	−40/+85°C	4.0 MHz	
Z84C00A F6	Frit Seal	−40/+85°C		
Z84C00A C6	Plastic Chip-Carrier	−40/+85°C		
Z84C00A K6	Ceramic Chip-Carrier	−40/+85°C		

Packages (dimension in mm)

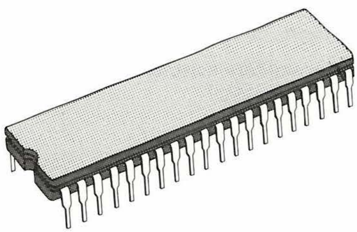
40 Lead Plastic DIP

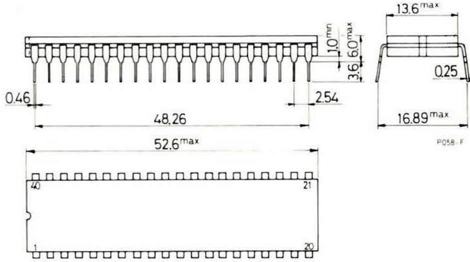




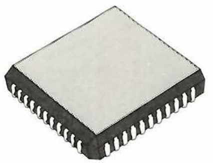
Packages (Continued)

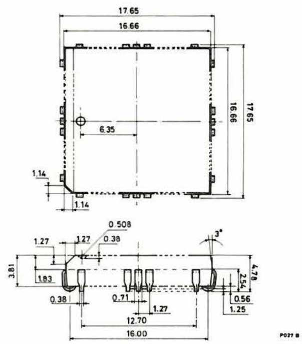
40 - Lead Ceramic DIP (Frit-Seal)



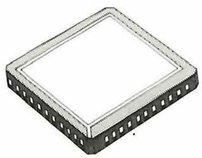


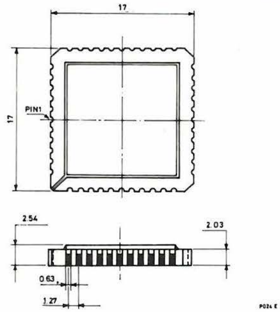
44 - Leaded Plastic Chip Carrier





44 - Leadless Ceramic Chip Carrier





Z80C PIO CMOS Version

Z84C20

Features

- Provides a direct interface between Z80* microcomputer systems and peripheral devices.
- Both ports have interrupt-driven handshake for fast response.
- Four programmable operating modes: byte input, byte output, byte input/output (Port A only), and bit input/output.
- Programmable interrupts on peripheral status conditions.
- Standard Z80 Family bus-request and prioritized interrupt-request daisy chains implemented without external logic.
- The eight Port B outputs can drive Darlington transistors (1.5 mA at 1.5 V).
- Single 5 V ± 10% Power Supply.
- Low power consumption: 2 mA typ at 4 MHz Less than 10 μA in Power Down mode.
- Extended Operating Temperature: −40°C to +85°C

General Description

The Z80C PIO Parallel I/O Circuit is a programmable, dual-port device that provides a TTL-compatible interface between peripheral devices and the CPU. The CPU configures the PIO to interface with a wide range of peripheral devices with no other external logic. Typical peripheral devices that are compatible with the PIO include most keyboards, paper tape readers and punches, printers, PROM programmers, etc.

One characteristic of the Z80 peripheral controllers that separates them from other interface controllers is that all data transfer between the peripheral device and the CPU is accomplished under interrupt control. Thus, the interrupt logic of the PIO permits full use of the efficient interrupt capabilities of the CPU during I/O transfers. All logic necessary to implement a fully nested interrupt structure is included in the PIO.

Another feature of the PIO is the ability to interrupt the CPU upon occurrence of specified status conditions in the peripheral device. For example, the PIO can be programmed to interrupt if any specified peripheral alarm conditions should occur. This interrupt capability reduces the time the processor must spend in polling peripheral status.

The Z80C PIO interfaces to peripherals via two independent general-purpose I/O ports, designated Port A and Port B. Each port has eight data bits and two handshake signals, Ready and Strobe, which control data transfer. The Ready output indicates to the peripheral that the port is ready for a data transfer. Strobe is an input from the peripheral that indicates when a data transfer has occured.

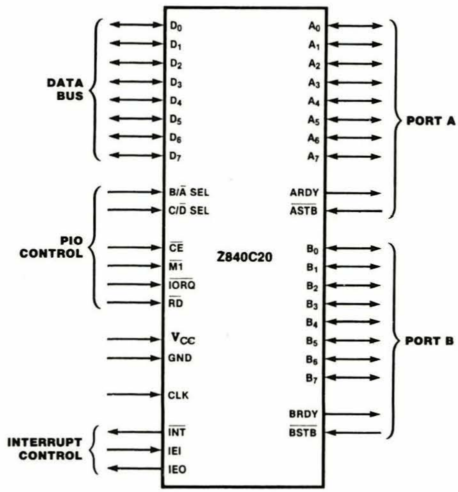


Figure 1. Logic Functions

General Description (Continued)

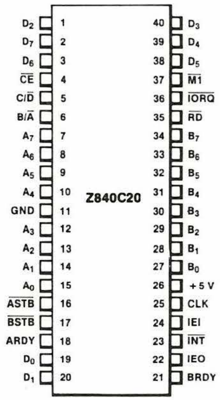


Figure 2. Pin Configuration

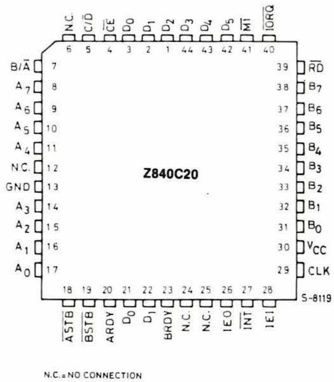


Figure 2a. Chip Carrier Pin Configuration

Operating Modes. The PIO ports can be programmed to operate in four modes: byte output (Mode 0), byte input (Mode 1), byte input/output (Mode 2) and bit input/output (Mode 3).

In Mode 0, either Port A or Port B can be programmed to output data. Both ports have output registers that are individually addressed by the CPU; data can be written to either port at any time. When data is written to a port, an active Ready output indicates to the external device that data is available at the associated port and is ready for transfer to the external device. After the data transfer, the external device responds with an active Strobe input, which generates an interrupt, if enabled.

In Mode 1, either Port A or Port B can be configured in the input mode. Each port has an input register addressed by the CPU. When the CPU reads data from a port, the PIO sets the Ready signal, which is detected by the external device. The external device then places data on the I/O lines and strobes the I/O port, which latches the data into the Port Input Register, resets Ready, and triggers the Interrupt Request, if enabled. The CPU can read the input data at any time, which again sets Ready.

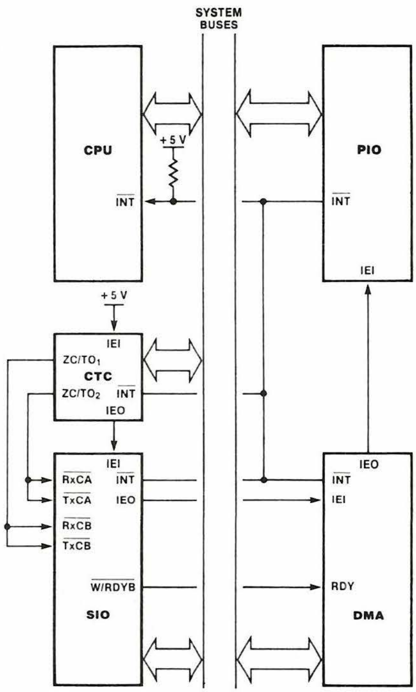


Figure 3. PIO in a Typical Z80C Family Environment

General Description (Continued)

Mode 2 is bidirectional and uses Port A, plus the interrupts and handshake signals from both ports. Port B must be set to Mode 3 and masked off. In operation, Port A is used for both data input and output. Output operation is similar to Mode 0 except that data is allowed out onto the Port A bus only when $\overline{ASTB}$ is Low. For input, operation is similar to Mode 1, except that the data input uses the Port B handshake signals and the Port B interrupt (if enabled).

Both ports can be used in Mode 3. In this mode, the individual bits are defined as either input or output bits. This provides up to eight separate, individually defined bits for each port. During operation, Ready and Strobe are not used. Instead, an interrupt is generated if the condition of one input changes, or if all inputs change. The requirements for generating an interrupt are defined during the programming operation; the active level is specified as either High or Low, and the logic condition is specified as either one input active (OR) or all inputs active (AND). For example, if the port is programmed for active Low inputs and the logic function is AND, then all inputs at the specified port must go Low to generate an interrupt.

Data outputs are controlled by the CPU and can be written or changed at any time.

- Individual bits can be masked off.
- The handshake signals are not used in Mode 3, Ready is held Low, and Strobe is disabled.
- When using PIO interrupts, the CPU interrupt mode must be set to Mode 2.

Internal Structure

The internal structure of the Z80C PIO consists of a CPU bus interface, internal control logic, Port A I/O logic, Port B I/O logic, and interrupt control logic (Figure 4). The CPU bus interface logic allows the PIO to interface directly to the CPU with no other external logic. The internal control logic synchronizes the CPU data bus to the peripheral device interfaces (Port A and Port B). The two I/O ports (A and B) are virtually identical and are used to interface directly to peripheral devices.

Port Logic. Each port contains separate input and output registers, handshake control logic, and the control registers shown in Figure 5. All data transfers between the peripheral unit and the CPU use the data

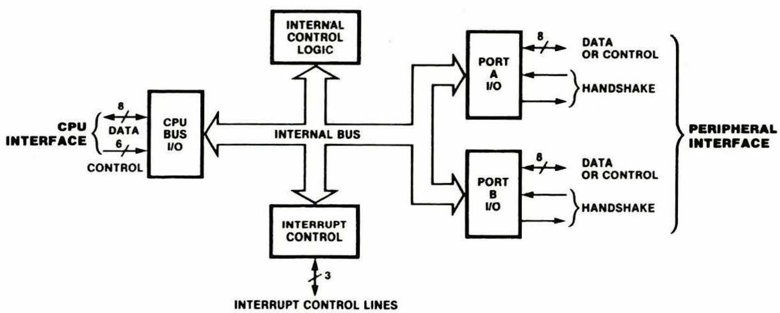


Figure 4. Block Diagram

Internal Structure (Continued)

input and output registers. The handshake logic associated with each port controls the data transfers through the input and the output registers. The mode control register (two bits) selects one of the four programmable operating modes.

The control mode (Mode 3) uses the remaining registers. The input/output control register specifies which of the eight data bits in the port are to be outputs and enables these bits; the remaining bits are inputs. The mask register and the mask control register control Mode 3 interrupt conditions. The mask register specifies which of the bits in the port are active and which are masked or inactive.

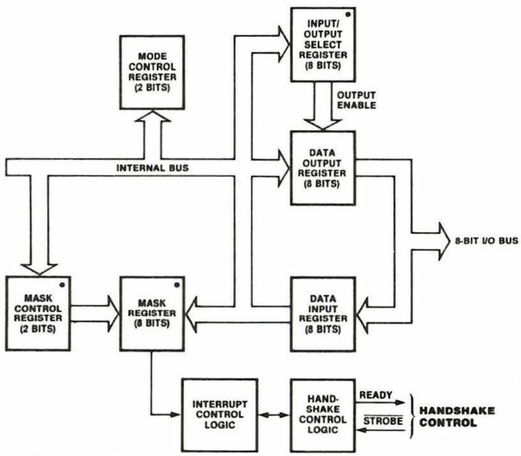
The mask control register specifies two conditions: first, whether the active state of the input bits is High or Low, and second, whether an interrupt is generated when any *one* unmasked input bit is active (OR condition) or if the interrupt is generated when *all* unmasked input bits are active (AND condition).

Interrupt Control Logic. The interrupt control logic section handles all CPU interrupt protocol for nested-priority interrupt structures. Any device's physical location in a daisy-chain configuration determines its priority. Two lines (IEI and IEO) are provided in each PIO to form this daisy chain. The device closest to the CPU has the highest priority. Within a PIO, Port A interrupts have higher priority than those of Port B. In the byte input, byte output, or bidirectional modes, an interrupt can be generated whenever the peripheral requests a new byte transfer. In the bit control mode, an interrupt can be generated when the peripheral status matches a programmed value. The PIO provides for complete control of nested interrupts. That is, lower priority devices may not interrupt higher priority devices that have not had their interrupt service routines completed by the CPU. Higher priority devices may interrupt the servicing of lower priority devices.

If the CPU (in interrupt Mode 2) accepts an interrupt, the interrupting device must provide an 8-bit interrupt vector for the CPU. This vector forms a pointer to a location in memory where the address of the interrupt service routine is located. The 8-bit vector from the interrupting device forms the least significant eight bits of the indirect pointer while the I Register in the CPU provides the most significant eight bits of the pointer. Each port (A and B) has an independent interrupt vector. The least significant bit of the vector is automatically set to 0 within the PIO because the pointer must point to two adjacent memory locations for a complete 16-bit address.

Unlike the other Z80C peripherals, the PIO does not enable interrupts immediately after programming. It waits until $\overline{M1}$ goes Low (e.g., during an opcode fetch). This condition is unimportant in the Z80C environment but might not be if another type of CPU is used.

The PIO decodes the RETI (Return From Interrupt) instruction directly from the CPU data bus so that each PIO in the system knows at all times whether it is being



*Used in the bit mode only to allow generation of an interrupt if the peripheral I/O pins go to the specified state.

Figure 5. Typical Port I/O Block Diagram

Internal Structure (Continued)

serviced by the CPU interrupt service routine. no other communication with the CPU is required.

CPU Bus I/O Logic. The CPU bus interface logic interfaces the PIO directly to the CPU, so no external logic is necessary.
For large systems, however, address decoders and/or buffers may be necessary.

Internal Control Logic. This logic receives the control words for each port during programming and, in turn, controls the operating functions of the PIO. The control logic synchronizes the port operations, controls the port mode, port addressing, selects the read/write function, and issues appropriate commands to the ports and the interrupt logic. The PIO does not receive a write input from the CPU; instead, the $\overline{RD}$, $\overline{CE}$, $\overline{C/D}$ and $\overline{IORQ}$ signals generate the write input internally.

Programming

Mode 0, 1, or 2. *(Byte Input, Output, or Bidirectional).* Programming a port for Mode 0, 1, or 2 requires two words per port. These words are:

A Mode Control Word. Selects the port operating mode (Figure 6). This word may be written any time.

An Interrupt Vector. The Z80C PIO is designed for use with the Z80C CPU in interrupt Mode 2 (Figure 7). When interrupts are enabled, the PIO must provide an interrupt vector.

Mode 3. *(Bit Input/Output).* Programming a port for Mode 3 operation requires a control word, a vector (if interrupts are enabled), and three additional words, described as follows:

I/O Register Control. When Mode 3 is selected, the mode control word must be followed by another control word that sets the I/O control register, which in turn defines which port lines are inputs and which are outputs (Figure 8).

Interrupt Control Word. In Mode 3, handshake is not used. Interrupts are generated as a logic function of the input signal levels. The interrupt control word sets the logic conditions and the logic levels required for generating an interrupt. Two logic conditions or functions are available: AND (if all input bits change to the active level, an interrupt is triggered), and OR (if any one of the input bits changes to the active level, an interrupt is triggered). Bit D_6 sets the logic function, as shown in Figure 9. The active level of the input bits can be set either High or Low. The active level is controlled by Bit D_5.

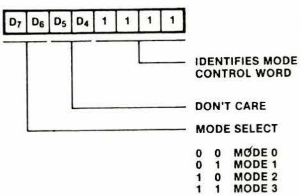


Figure 6. Mode Control Word

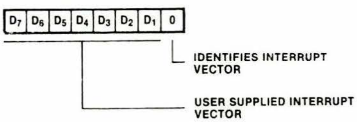


Figure 7. Interrupt Vector Word

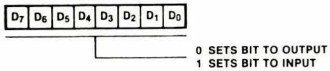
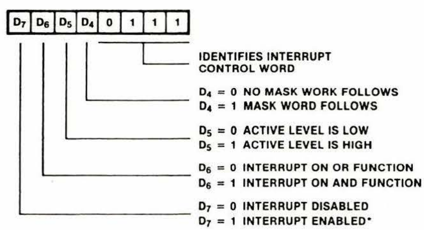


Figure 8. I/O Register Control Word



*NOTE: THE PORT IS NOT ENABLED UNTIL THE INTERRUPT ENABLE IS FOLLOWED BY AN ACTIVE $\overline{M1}$.

Figure 9. Interrupt Control Word

Programming (Continued)

Mask Control Word. This word sets the mask control register, allowing any unused bits to be masked off. If any bits are to be masked, then D_4 must be set. When D_4 is set, the next word written to the port must be a mask control word (Figure 10).

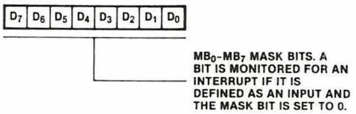


Figure 10. Mask Control Word

Interrupt Disable. There is one other control word which can be used to enable or disable a port interrupt. It can be used without changing the rest of the interrupt control word. (Figure 11).

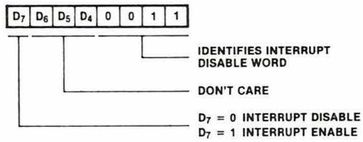


Figure 11. Interrupt Disable Word

Pin Descritpion

A_0-A_7. *Port A Bus* (bidirectional, 3-state). This 8-bit bus transfers data, status, or control information between Port A of the PIO and a peripheral device. A_0 is the least significant bit of the Port A data bus.

ARDY. *Register A Ready* (output, active High)., The meaning of this signal depends on the mode of operation selected for Port A as follows:

Output Mode. This signal goes active to indicate that the Port A output register has been loaded and the peripheral data bus is stable and ready for transfer to the peripheral device.

Input Mode. This signal is active when the Port A input register is empty and ready to accept data from the peripheral device.

Bidirectional Mode. This signal is active when data is available in the Port A output register for transfer to the peripheral device. In this mode, data is not placed on the Port A data bus, unless $\overline{\text{ASTB}}$ is active.

Control Mode. This signal is disabled and forced to a Low state.

$\overline{\text{ASTB}}$. *Port A Strobe Pulse From Peripheral Device* (input, active Low). The meaning of this signal depends on the mode of operation selected for Port A as follows:

Output Mode. The positive edge of this strobe is issued by the peripheral to acknowledge the receipt of data made available by the PIO.

Input Mode. The strobe is issued by the peripheral to load data from the peripheral into the Port A input register. Data is loaded into the PIO when this signal is active.

Bidirectional Mode. When this signal is active, data from the Port A output register is gated onto the Port A bidirectional data bus. The positive edge of the strobe acknowledges the receipt of the data.

Control Mode. The strobe is inhibited internally.

B_0-B_7. *Port B Bus* (bidirectional, 3-state). This 8-bit bus transfers data, status, or control information between Port B and a peripheral device. The Port B data bus can supply 1.5 mA at 1.5 V to drive Darlington transistors. B_0 is the least significant bit of the bus.

B/$\overline{\text{A}}$. *Port B Or A Select* (input, High = B). This pin defines which port is accessed during a data transfer between the CPU and the PIO. A Low on this pin selects Port A; a High selects Port B. Often address bit A_0 from the CPU is used for this selection function.

BRDY. *Register B Ready* (output, active High). This signal is similar to ARDY, except that in the Port A bidirectional mode this signal is High when the Port A input register is empty and ready to accept data from the peripheral device.

Pin Description (Continued)

$\overline{BSTB}$. *Port B Strobe Pulse From Peripheral Device* (input, active Low). This signal is similar to $\overline{ASTB}$, except that in the Port A bidirectional mode this signal strobes data from the peripheral device into the Port A input register.

C/$\overline{D}$. *Control Or Data Select* (input, High = C). This pin defines the type of data transfer to be performed between the CPU and the PIO. A High on this pin during a CPU write to the PIO causes the data bus to be interpreted as a *command* for the port selected by the B/$\overline{A}$ Select line. A Low on this pin means that the data bus is being used to transfer data between the CPU and the PIO. Often address bit A_1 from the CPU is used for this function.

$\overline{CE}$. *Chip Enable* (input, active Low). A Low on this pin enables the PIO to accept command or data inputs from the CPU during a write cycle or to transmit data to the CPU during a read cycle. This signal is generally decoded from four I/O port numbers for Ports A and B, data, and control.

CLK. *System Clock* (input). The Z80C PIO uses the standard single-phase Z80C system clock.

D_0-D_7. *CPU Data Bus* (bidirectional, 3-state). This bus is used to transfer all data and commands between the CPU and the PIO. D_0 is the least significant bit.

IEI. *Interrupt Enable In* (input, active High). This signal is used to form a priority-interrupt daisy chain when more than one interrupt-driven device is being used. A High level on this pin indicates that no other devices of higher priority are being serviced by a CPU interrupt service routine.

IEO. *Interrupt Enable Out* (output, active High). The IEO signal is the other signal required to form a daisy chain priority scheme. It is High only if IEI is High and the CPU is not servicing an interrupt from this PIO. Thus this signal blocks lower priority devices from interrupting while a higher priority device is being serviced by its CPU interrupt service routine.

$\overline{INT}$. *Interrupt Request* (output, open drain, active Low). When $\overline{INT}$ is active the PIO is requesting an interrupt from the CPU.

$\overline{IORQ}$. *Input/Output Request* (input from CPU, active Low). $\overline{IORQ}$ is used in conjunction with B/$\overline{A}$, C/$\overline{D}$, $\overline{CE}$, and $\overline{RD}$ to transfer commands and data between the CPU and the PIO. When $\overline{CE}$, $\overline{RD}$, and $\overline{IORQ}$ are active, the port addressed by B/$\overline{A}$ transfers data to the CPU (a read operation). Conversely, when $\overline{CE}$ and $\overline{IORQ}$ are active but $\overline{RD}$ is not, the port addressed by B/$\overline{A}$ is written into from the CPU with either data or control information, as specified by C/$\overline{D}$. Also, if $\overline{IORQ}$ and $\overline{M1}$ are active simultaneously, the CPU is acknowledging an interrupt; the interrupting port automatically places its interrupt vector on the CPU data bus if it is the highest priority device requesting an interrupt.

$\overline{M1}$. *Machine Cycle* (input from CPU, active Low). This signal is used as a sync pulse to control several internal PIO operations. When both the $\overline{M1}$ and $\overline{RD}$ signals are active, the CPU is fetching an instruction from memory. Conversely, when both $\overline{M1}$ and $\overline{IORQ}$ are active, the CPU is acknowledging an interrupt. In addition, $\overline{M1}$ has two other functions within the PIO: it synchronizes the PIO interrupt logic; when $\overline{M1}$ occurs without an active $\overline{RD}$ or $\overline{IORQ}$ signal, the PIO is reset.

$\overline{RD}$. *Read Cycle Status* (input from CPU, active Low). If $\overline{RD}$ is active, or an I/O operation is in progress, $\overline{RD}$ is used with B/$\overline{A}$, C/$\overline{D}$, $\overline{CE}$, and $\overline{IORQ}$ to transfer data from the PIO to the CPU.

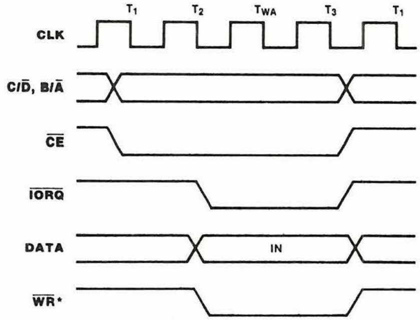
Timing

The following timing diagrams show typical timing in a Z80C CPU environment. For more precise specifications refer to the composite ac timing diagram.

Write Cycle. Figure 12 illustrates the timing for programming the Z80C PIO or for writing data to one of its ports. No Wait states are allowed for writing to the PIO other than the automatically inserted T_{WA}. The PIO does not receive a specific write signal; it internally generates its own from the lack of an active $\overline{RD}$ signal.

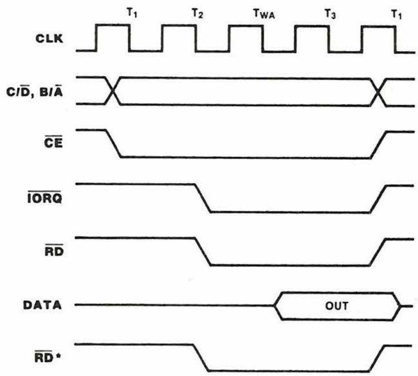
Read Cycle. Figure 13 illustrates the timing for reading the data input from an external device to one of the PIO ports. No Wait states are allowed for reading the PIO other than the automatically inserted T_{WA}.

Output Mode (Mode 0). An output cycle (Figure 14) is always started by the execution of an output instruction by the CPU. The $\overline{WR}^*$ pulse from the CPU latches the data from the CPU data bus into the selected port's output register. The $\overline{WR}^*$ pulse sets the Ready flag after a Low-going edge of CLK, indicating data is available. Ready stays active until the positive edge of the strobe line is received, indicating that data was taken by the peripheral. The positive edge of the strobe pulse generates an $\overline{INT}$ if the interrupt enable flip-flop has been set and if this device has the highest priority.



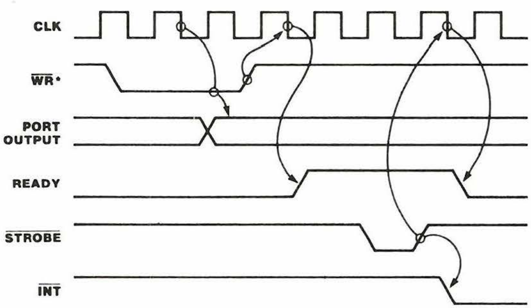
$^*\overline{WR} = RD \bullet \overline{CE} \bullet \overline{C/D} \bullet \overline{IORQ}$

Figure 12. Write Cycle Timing



$^*\overline{RD} = \overline{RD} \bullet \overline{CE} \bullet \overline{C/D} \bullet \overline{IORQ}$

Figure 13. Read Cycle Timing



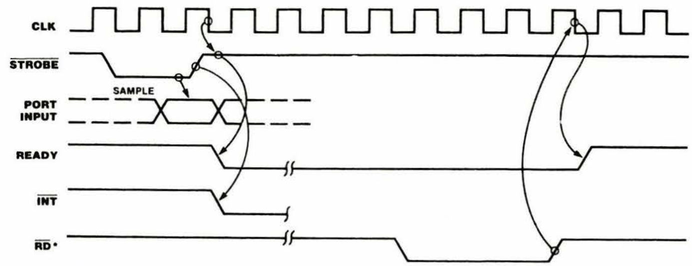
$^*\overline{WR} = RD \bullet \overline{CE} \bullet \overline{C/D} \bullet \overline{IORQ}$

Figure 14. Mode 0 Output Timing

Timing (Continued)

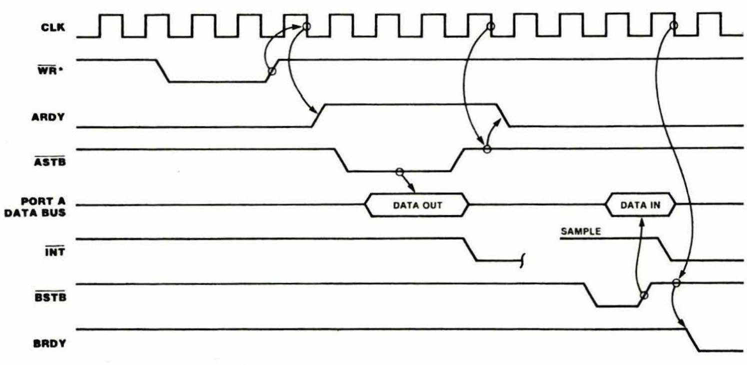
Input Mode (Mode 1). When $\overline{STROBE}$ goes Low, data is loaded into the selected port input register (Figure 15). The next rising edge of strobe activates $\overline{INT}$, if Interrupt Enable is set and this is the highest-priority requesting device. The following falling edge of CLK resets Ready to an inactive state, indicating that the input register is full and cannot accept any more data until the CPU completes a read. When a read is complete, the positive edge of $\overline{RD}$ sets Ready at the next Low-going transition of CLK. At this time new data can be loaded into the PIO.

Bidirectional Mode (Mode 2). This is a combination of Modes 0 and 1 using all four handshake lines and the eight Port A I/O lines (Figure 16). Port B must be set to the bit mode and its inputs must be masked. The Port A handshake lines are used for output control and the Port B lines are used for input control. If interrupts occur, Port A's vector will be used during port output and Port B's will be used during port input. Data is allowed out onto the Port A bus only when $\overline{ASTB}$ is Low. The rising edge of this strobe can be used to latch the data into the peripheral.



$^{*}\overline{RD} = \overline{RD} \bullet \overline{CE} \bullet \overline{C/D} \bullet \overline{IORQ}$

Figure 15. Mode 1 Input Timing



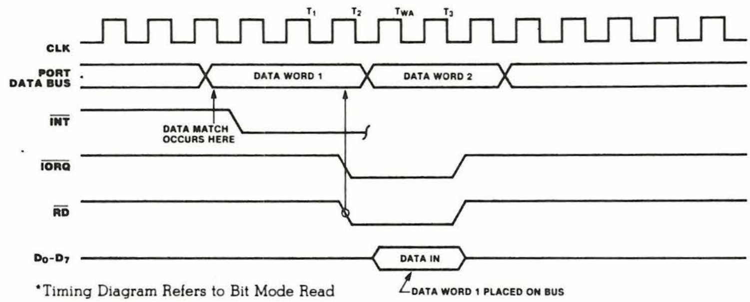
$^{*}\overline{WR} = RD \bullet \overline{CE} \bullet \overline{C/D} \bullet \overline{IORQ}$

Figure 16. Mode 2 Bidirectional Timing

Timing (Continued)

Bit Mode (Mode 3). The bit mode does not utilize the handshake signals, and a normal port write or port read can be executed at any time. When writing, the data is latched into the output registers with the same timing as the output mode (Figure 17).

When reading the PIO, the data returned to the CPU is composed of output register data from those port data lines assigned as outputs and input register data from those port data lines assigned as inputs. The input register contains data that was present immediately prior to the falling edge of $\overline{RD}$. An interrupt is generated if interrupts from the port are enabled and the data on the port data lines satisfy the logical equation defined by the 8-bit mask and 2-bit mask control registers. However, if Port A is programmed in bidirectional mode, Port B does not issue an interrupt in bit mode and must therefore be polled.



*Timing Diagram Refers to Bit Mode Read

Figure 17. Mode 3 Bit Mode Timing

Interrupt Acknowledge Timing. During $\overline{M1}$ time, peripheral controllers are inhibited from changing their interrupt enable status, permitting the Interrupt Enable signal to ripple through the daisy chain. The peripheral with IEI High and IEO Low during $\overline{INTACK}$ places a preprogrammed 8-bit interrupt vector on the data bus at this time (Figure 18). IEO is held Low until a Return From Interrupt (RETI) instruction is executed by the CPU while IEI is High. The 2-byte RETI instruction is decoded internally by the PIO for this purpose.

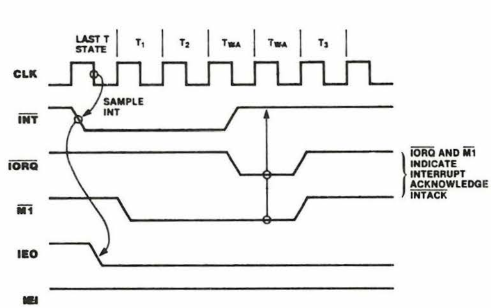


Figure 18. Interrupt Acknowledge Timing

Timing (Continued)

Return From Interrupt Cycle. If a Z80C peripheral has no interrupt pending and is not under service, then its IEO = IEI. If it has an interrupt under service (i.e., it has already interrupted and received an interrupt acknowledge) then its IEO is always Low, inhibiting lower priority devices from interrupting. If it has an interrupt pending which has not yet been acknowledged, IEO is Low unles an "ED" is decoded as the first byte of a 2-byte opcode (Figure 19). In this case, IEO goes High until the next opcode byte is decoded, whereupon it goes Low again. If the second byte of the opcode was a "4D", then the opcode was an RETI instruction.

After an "ED" opcode is decoded, only the peripheral device which has interrupted and is currently under service has its IEI High and its IEO Low. This device is the highest-priority device in the daisy chain that has received an interrupt acknowledge. All other peripherals have IEI = IEO. If the next opcode byte decoded is "4D", this peripheral device resets its "interrupt under service" condition.

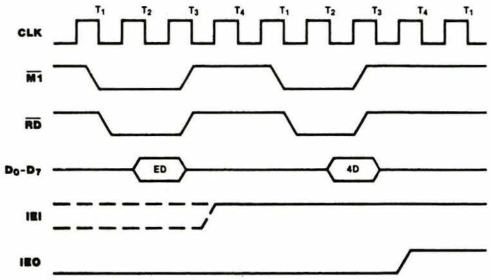
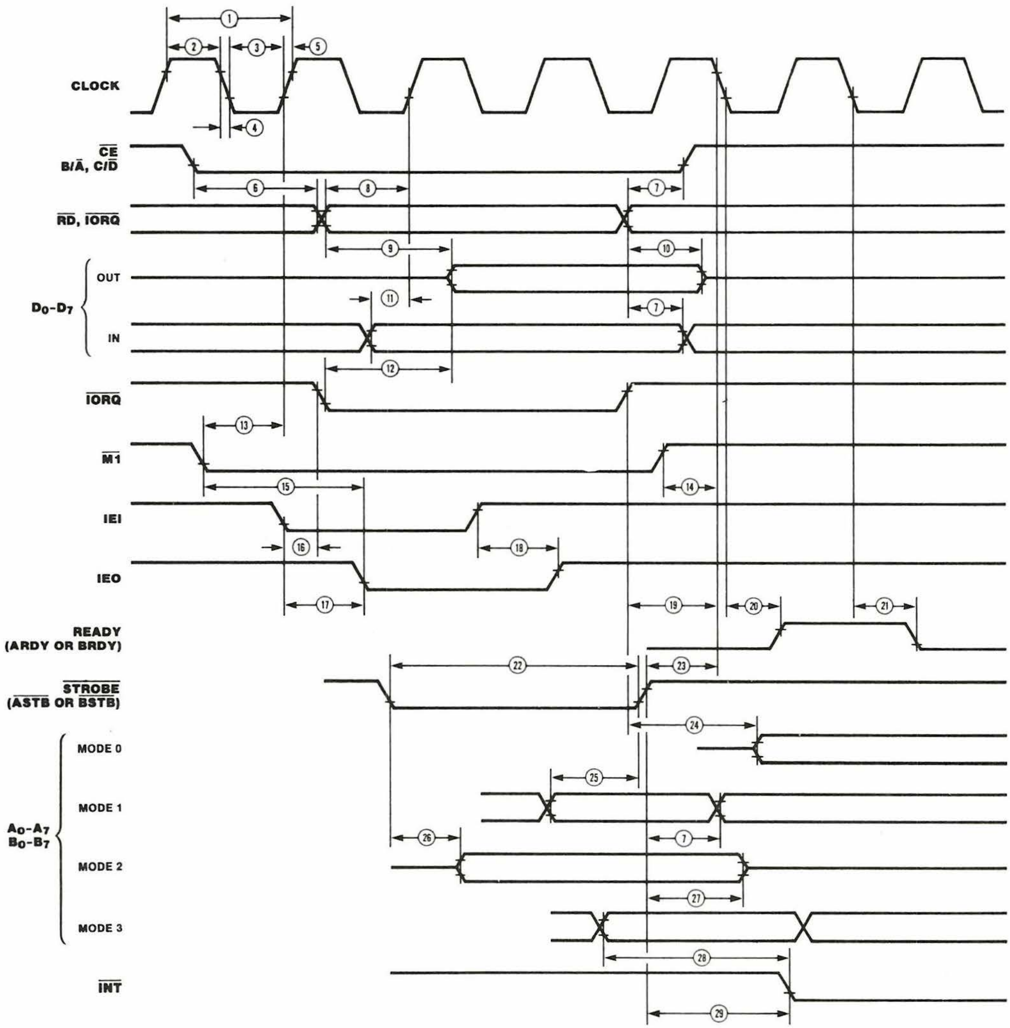


Figure 19. Return From Interrupt

AC Characteristics



AC Characteristics (Continued)

Number	Symbol	Parameter	Z84C20A Min (ns)	Z84C20A Max (ns)
1	TcC	Clock Cycle Time	250	—
2	TwCh	Clock Width (High)	105	—
3	TwCl	Clock Width (Low)	105	—
4	TfC	Clock Fall Time	—	30
5	TrC	Clock Rise Time	—	30
6	TsCS(RI)	$\overline{CE}$, B/$\overline{A}$, C/$\overline{D}$ to $\overline{RD}$, $\overline{IORQ}$ ↓ Setup Time	50	—
7	Th	Any Hold Times for Specified Setup Time	40	—
8	TsRI(C)	$\overline{RD}$, $\overline{IORQ}$ to Clock ↑ Setup Time	115	—
9	TdRI(DO)	$\overline{RD}$, $\overline{IORQ}$ ↓ to Data Out Delay	—	380
10	TdRI(DOs)	$\overline{RD}$, $\overline{IORQ}$ ↑ to Data Out Float Delay	—	110
11	TsDI(C)	Data In to Clock ↑ Setup Time	50	—
12	TdIO(DOI)	$\overline{IORQ}$ ↓ to Data Out Delay (INTACK Cycle)	160	—
13	TsMI(Cr)	$\overline{MI}$ ↓ to Clock ↑ Setup Time	90	—
14	TsMI(Cf)	$\overline{MI}$ ↑ to Clock ↓ Setup Time ($\overline{MI}$ Cycle)	0	—
15	TdMI(IEO)	$\overline{MI}$ ↓ to IEO ↓ Delay (Interrupt Immediately Preceding $\overline{MI}$ ↓)	—	190
16	TsIEI(IO)	IEI ↓ to $\overline{IORQ}$ ↓ Setup Time (INTACK Cycle)	140	—
17	TdIEI(IEOf)	IEI ↓ to IEO ↓ Delay	—	130
18	TdIEI(IEOr)	IEI ↑ to IEO ↑ Delay (after ED Decode)	—	160
19	TcIO(C)	$\overline{IORQ}$ ↑ to Clock ↓ Setup Time (To Activate READY on Next Clock Cycle)	200	—
20	TdC(RDYr)	Clock ↓ to READY ↑ Delay	—	190
21	TdC(RDYf)	Clock ↓ to READY ↓ Delay	—	140
22	TwSTB	$\overline{STROBE}$ Pulse Width	150	—
23	TsSTB(C)	$\overline{STROBE}$ ↑ to Clock ↓ Setup Time (To Activate READY on Next Clock Cycle)	220	—
24	TdIO(PD)	$\overline{IORQ}$ ↑ to PORT DATA Stable Delay (Mode 0)	—	180
25	TsPD(STB)	PORT DATA to $\overline{STROBE}$ ↑ Setup Time (Mode 1)	230	—
26	TdSTB(PD)	$\overline{STROBE}$ ↓ to PORT DATA Stable (Mode 2)	—	210
27	TdSTB(PDr)	$\overline{STROBE}$ ↑ to PORT DATA Float Delay (Mode 2)	—	180
28	TdPD(INT)	PORT DATA Match to $\overline{INT}$ ↓ Delay (Mode 2)	—	490
29	TdSTB(INT)	$\overline{STROBE}$ ↑ to $\overline{INT}$ ↑ Delay	—	440

Absolute Maximum Ratings

Symbol	Item	Rating
V_{CC}	V_{CC} Supply Voltage with respect to V_{SS}	−0.5 to 7V
V_{IN}	Input Voltage	−0.5V to V_{CC}+0.5V
P_D	Power Dissipation (T_A=85°C)	250 mW
T_{SOLDER}	Soldering Temperature (Soldering Time 10 sec)	260°C
T_{STG}	Storage Temperature	−65°C to 150°C
T_{OPR}	Operating Temperature	−40°C to 85°C

DC Characteristics (1)

Symbol	Parameter	Min.	Typ.	Max.	Unit	Test Conditions
V_{ILC}	Clock Input Low Voltage	−0.3	—	0.6	V	
V_{IHC}	Clock Input High Voltage	V_{CC} −0.6	—	V_{CC} +0.3	V	
V_{IL}	Input Low Voltage (except CLK)	−0.5	—	0.8	V	
V_{IH}	Input High Voltage (except CLK)	2.2	—	V_{CC}	V	
V_{OL}	Output Low Voltage	—	—	0.4	V	I_{OL}=2.0mA
V_{OH1}	Output High Voltage (1)	2.4	—	—	V	I_{OH}=−1.6mA
V_{OH2}	Output High Voltage (2)	V_{CC} −0.8	—	—	V	I_{OH}=−250μA
I_{LI}	Input Leakage Current	—	—	±10	μA	$V_{SS} \leq V_{IN} \leq V_{CC}$
I_{OL}	3-State Output Leakage Current in Float	—	—	10	μA	$V_{SS}+0.4 \leq V_{OUT} \leq V_{CC}$
I_{CC1}	Operating Supply Current	—	2	5	mA	V_{CC}=5V, CLK=4MHz $V_{IH}=V_{CC}-0.2V$ V_{IL}=0.2V
I_{CC2}	Stand-by Supply Current	—	0.5	10	μA	V_{CC}=5V CLK=V_{CC} $V_{IH}=V_{CC}-0.2V$ V_{IL}=0.2V
*I_{OHD}	Darlington Drive Current	−1.5	—	−5.0	mA	V_{OH}=1.5V, R_{EXT}=1.1kΩ

Note (1): * Applied to Port B only

(2) Typical value is specified at 25°C

Test Conditions

T_A = −40°C to +85°C
V_{CC}= 5V ±10%
V_{SS}=0V

AC test conditions

- Inputs except CLK (clock) are driven at 2.4V for a logic "1" and 0.4V for a logic "0". Clock input is driven at $V_{CC}-0.6V$ for a logic "1" and 0.6V for a logic "0".
- Timing measurements are made at 2.2V for a logic "1" and 0.8V for a logic "0".

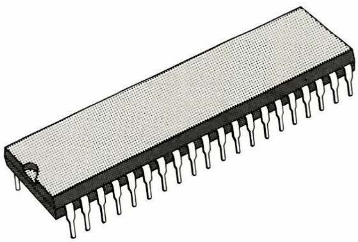
All AC parameters assume a load capacitance of 100pF

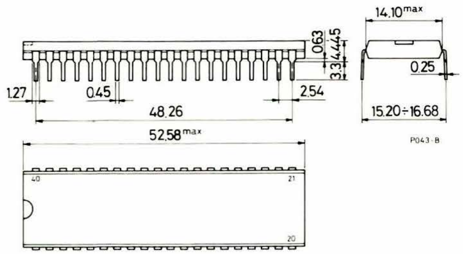
Ordering Information

Type	Package	Temp.	Clock	Description
Z84C20A B6	Plastic	– 40/ + 85°C	4.0 MHz	Z80C Parallel I/O Unit CMOS Version
Z84C20A F6	Frit Seal	– 40/ + 85°C		
Z84C20A C6	Plastic Chip Carrier	– 40/ + 85°C		
Z84C20A K6	Ceramic Chip Carrier	– 40/ + 85°C		

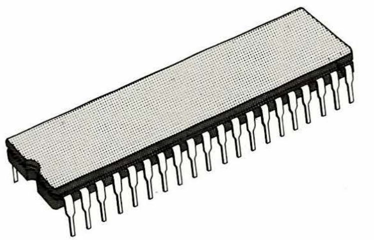
Packages (dimension in mm)

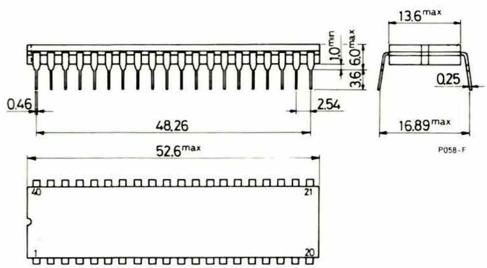
40 - Lead Plastic DIP





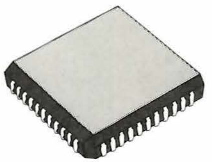
40 - Lead Ceramic DIP (Frit-Seal)

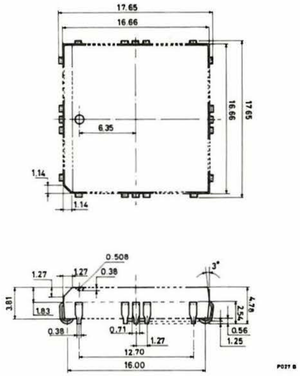




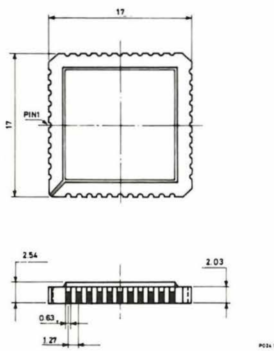
Packages (Continued)

44 - Leaded Plastic Chip Carrier





44 - Leadless Ceramic Chip Carrier



Z80C CTC
CMOS Version

Z84C30

Features

- Four independently programmable counter/timer channels, each with a readable downcounter and a selectable 16 or 256 prescaler. Downcounters are reloaded automatically at zero count.
- Three channels have Zero Count/Timeout outputs capable of driving Darlington transistors.
- Selectable positive or negative trigger initiates timer operation.
- Standard Z80C Family daisy-chain interrupt structure provides fully vectored, prioritized interrupts without external logic. The CTC may also be used as an interrupt controller.
- Interface directly to the Z80C CPU or-for baud rate generation - to the Z80C SIO.
- Single 5 V ± 10% Power Supply.
- Low Power Consumption:
  3 mA tip. at 4 MHz.
  Less than 10 μA in Power Down mode.
- Extended Operating Temperature:
  −40°C to +85°C.

General Description

The Z80C CTC four-channel counter/timer can be programmed by system software for a broad range of counting and timing applications. The four independently programmable channels of the CTC satisfy common microcomputer system requirements for event counting, interrupt and interval timing, and general clock rate generation.

System design is simplified because the CTC connects directly to both the CPU and the SIO with no additional logic. In larger systems, address decoders and buffers may be required.

Programming the CTC is straightforward: each channel is programmed with two bytes; a third is necessary when interrupts are enabled. Once started, the CTC counts down, reloads its time constant automatically, and resumes counting. Software timing loops are completely eliminated. Interrupt processing is simplified because only one vector need be specified; the CTC internally generates a unique vector for each channel.

The Z80C CTC requires a single +5 V power supply and the standard Z80C single-phase system clock. It is fabricated with n-channel silicon-gate depletion-load technology, and packaged in a 28-pin plastic or ceramic DIP.

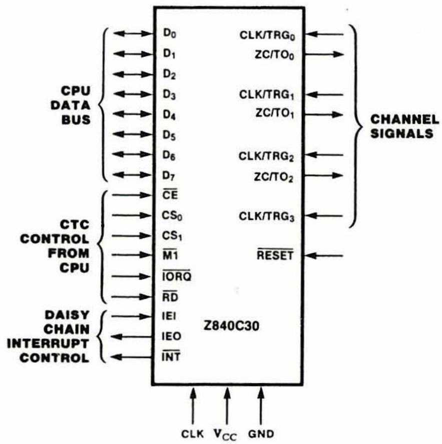


Figure 1. Logic Functions

General Description (Continued)

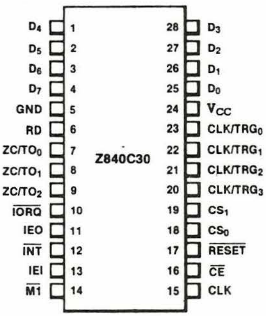


Figure 2. Pin Configuration

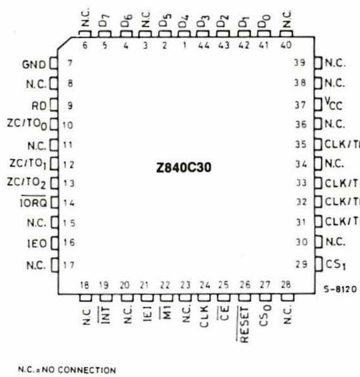


Figure 2a. Chip-Carrier Pin Configuration

Functional Description

The Z80C CTC has four independent counter/timer channels. Each channel is individually programmed with two words: a control word and a time-constant word. The control word selects the operating mode (counter or timer), enables or disables the channel interrupt, and selects certain other operating parameters. If the timing mode is selected, the control word also sets a prescaler, which divides the system clock by either 16 or 256. The time-constant word is a value from 1 to 256.

During operation, the individual counter channel counts down from the preset time constant value. In counter mode operation the counter decrements on each of the CLK/TRG input pulses until zero count is reached. Each decrement is synchronized by the system clock. For counts greater than 256, more than one counter can be cascaded. At zero count, the down-counter is automatically reset with the time constant value.

The timer mode determines time interval as small as 4 μs (Z80CA) without additional logic or software timing loops. Time intervals are generated by dividing the system clock with a prescaler that decrements a preset down-counter.

Thus, the time interval is an integral multiple of the clock period, the prescaler value (16 or 256) and the time constant that is preset in the down-counter. A timer is triggered automatically when its time constant value is programmed, or by an external CLK/TRG input.

Three channels have two outputs that occur at zero count. The first output is a zero-count/timeout pulse at the ZC/TO output. The fourth channel (Channel 3) does not have a ZC/TO output; interrupt request is the only output available from Channel 3.

The second output is Interrupt Request ($\overline{\text{INT}}$), which occurs if the channel has its interrupt enabled during programming. When CPU acknowledges Interrupt Request, the CTC places an interrupt vector on the data bus.

The four channels of the CTC are fully prioritized and fit into four contiguous slots in a standard Z80C daisy-chain interrupt structure. Channel 0 is the highest priority and Channel 3 the lowest. Interrupts can be individually enabled (or disabled) for each of the four channels.

Architecture

The CTC has four major elements, as shown in Figure 3.

- CPU bus I/O
- Channel control logic
- Interrupt logic
- Counter/timer circuits

CPU Bus I/O. The CPU bus I/O circuit decodes the address inputs, and interfaces the CPU data and control signals to the CTC for distribution on the internal bus.

Internal Control Logic. The CTC internal control logic controls overall chip operating functions such as the chip enable, reset, and read/write logic.

Interrupt Logic. The interrupt control logic ensures that the CTC interrupts interface properly with the Z80C CPU interrupt system. The logic controls the interrupt priority of the CTC as a function of the IEI signal. If IEI is High, the CTC has priority. During interrupt processing, the interrupt logic holds IEO Low, which inhibits the interrupt operation on lower priority devices. If the IEI input goes Low, priority is relinquished and the interrupt logic drives IEO Low.

If a channel is programmed to request an interrupt, the interrupt logic drives IEO Low at the zero count, and generates an $\overline{\text{INT}}$ signal to the CPU. When the CPU responds with interrupt acknowledge ($\overline{\text{M1}}$ and $\overline{\text{IORQ}}$), then the interrupt logic arbitrates the CTC internal priorities, and the interrupt control logic places a unique interrupt vector on the data bus.

If an interrupt is pending, the interrupt logic holds IEO Low. When the CPU issues a Return From Interrupt (RETI) instruction, each peripheral device decodes the first byte (ED_{16}). If the device has a pending interrupt, it raises IEO (High) for one $\overline{\text{M1}}$ cycle. This ensures that all lower priority devices can decode the entire RETI instruction and reset properly.

Counter/Timer Circuits. The CTC has four independent counter/timer circuits, each containing the logic shown in Figure 4.

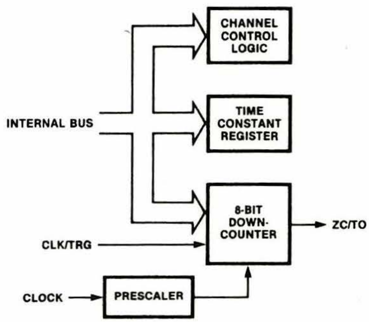


Figure 4. Counter/Timer Block Diagram

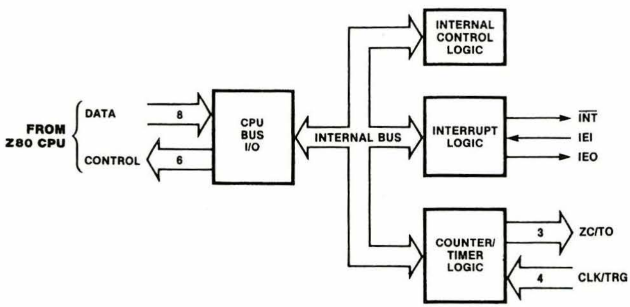


Figure 3. Functional Block Diagram

Architecture (Continued)

Channel Control Logic. The channel control logic receives the 8-bit channel control word when the counter/timer channel is programmed. The channel control logic decodes the control word and sets the following operating conditions:

- Interrupt enable (or disable)
- Operating mode (timer or counter)
- Timer mode prescaler factor (16 or 256)
- Active slope for CLK/TRG input
- Timer mode trigger (automatic or CLK/TRG input)
- Time constant data word to follow
- Software reset

Time Constant Register. When the counter/timer channel is programmed, the time constant register receives and stores an 8-bit time constant value, which can be anywhere from 1 to 256 (0 = 256). This constant is automatically loaded into the down-counter when the counter/time channel is initialized, and subsequently after each zero count.

Prescaler. The prescaler, which is used only in timer mode, divides the system clock frequency by a factor of either 16 or 256. The prescaler output clocks the down-counter during timer operation. The effect of the prescaler on the down-counter is a multiplication of the system clock period by 16 or 256. The prescaler factor is programmed by bit 5 of the channel control word.

Down-Counter. Prior to each count cycle, the down-counter is loaded with the time constant register contents. The counter is then decremented one of two ways, depending on operating mode:

- By the prescaler output (timer mode)
- By the trigger pulses into the CLK/TRG input (counter mode)

Without disturbing the down-count, the CPU can read the count remaining at any time by performing an I/O read operation at the port address assigned to the CTC channel. When the down-counter reaches the zero count, the ZC/TO output generates a positive-going pulse. When the interrupt is enabled, zero count also triggers an interrupt request signal ($\overline{\text{INT}}$) from the interrupt logic.

Programming

Each Z80C CTC channel must be programmed prior to operation. Programming consists of writing two words to the I/O port that corresponds to the desired channel. The first word is a control word that selects the operating mode and other parameters; the second word is a time constant, which is a binary data word with a value from 1 to 256. A time constant word must be preceded by a channel control word.

After initialization, channels may be reprogrammed at any time. If updated control and time constant words are written to a channel during the count operation, the count continues to zero before the new time constant is loaded into the counter.

If the interrupt on any CTC channel is enabled, the programming procedure should also include an interrupt vector. Only one vector is required for all four channels, because the interrupt logic automatically modifies the vector for the channel requesting service.

A control word is identified by a 1 in bit 0. A 1 in bit 2 indicates a time constant word is to follow. Interrupt vectors are always addressed to Channel 0, and identified by a 0 in bit 0.

Programming (Continued)

Addressing. During programming, channels are addressed with the channel select pins CS_1 and CS_2. A 2-bit binary code selects the appropriate channel as shown in the following table.

Channel	CS_1	CS_0
0	0	0
1	0	1
2	1	0
3	1	1

Reset. The CTC has both hardware and software resets. The hardware reset terminates all down-counts and disables all CTC interrupts by resetting the interrupt bits in the control registers. In addition, the ZC/TO and Interrupt outputs go inactive, IEO reflects IEI, and D_0-D_7 go to the high-impedance state. All channels must be completely reprogrammed after a hardware reset.

The software reset is controlled by bit 1 in the channel control word. When a channel receives a software reset, it stops counting. When a software reset is used, the other bits in the control word also change the contents of the channel control register. After a software reset a new time constant word must be written to the same channel.

If the channel control word has both bits D_1 and D_2 set to 1, the addressed channel stops operating, pending a new time constant word. The channel is ready to resume after the new constant is programmed. In timer mode, if $D_3 = 0$, operation is triggered automatically when the time constant word is loaded.

Channel Control Word Programming. The channel control word is shown in Figure 5. It sets the modes and parameters described below.

Interrupt Enable. D_7 enables the interrupt, so that an interrupt output ($\overline{INT}$) is generated at zero count. Interrupts may be programmed in either mode and may be enabled or disabled at any time.

Operating Mode. D_6 selects either timer or counter mode.

Prescaler Factor. (Timer Mode Only). D_5 selects factor—either 16 or 256.

Trigger Slope. D_4 selects the active edge or slope of the CLK/TRG input pulses. Note that reprogramming the CLK/TRG slope during operation is equivalent to issuing an active edge. If the trigger slope is changed by a control word update while a channel is pending operation in timer mode, the result is the same as a CLK/TRG pulse and the timer starts. Similarly, if the channel is in counter mode, the counter decrements.

Trigger Mode (Timer Mode Only). D_3 selects the trigger mode for timer operation. When

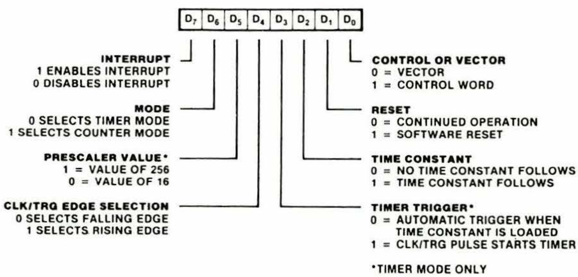


Figure 5. Channel Control Word

Programming (Continued)

D_3 is reset to 0, the timer is triggered automatically. The time constant word is programmed during an I/O write operation, which takes one machine cycle. At the end of the write operation there is a setup delay of one clock period. The timer starts automatically (decrements) on the rising edge of the second clock pulse (T_2) of the machine cycle following the write operation. Once started, the timer runs continuously. At zero count the timer reloads automatically and continues counting without interruption or delay, until stopped by a reset.

When D_3 is set to 1, the timer is triggered externally through the CLK/TRG input. The time constant word is programmed during an I/O write operation, which takes one machine cycle. The timer is ready for operation on the rising edge of the second clock pulse (T_2) of the following machine cycle. Note that the first timer decrement follows the active edge of the CLK/TRG pulse by a delay time of one clock cycle if a minimum setup time to the rising edge of clock is met. If this minimum is not met, the delay is extended by another clock period. Consequently, for immediate triggering, the CLK/TRG input must precede T_2 by one clock cycle plus its minimum setup time. If the minimum time is not met, the timer will start on the third clock cycle (T_3).

Once started the timer operates continuously, without interruption or delay, until stopped by a reset.

Time Constant to Follow. A 1 in D_2 indicates that the next word addressed to the selected channel is a time constant data word for the time constant register. The time constant word may be written at any time.

A 0 in D_2 indicates no time constant word is to follow. This is ordinarily used when the channel is already in operation and the new channel control word is an update. A channel will not operate without a time constant value. The only way to write a time **constant value is to write a control word with D_2 set.**

Software Reset. Setting D_1 to 1 causes a software reset, which is described in the Reset section.

Control Word. Setting D_0 to 1 identifies the word as a control word.

Time Constant Programming. Before a channel can start counting it must receive a time constant word from the CPU. During programming or reprogramming, a channel control word in which bit 2 is set must precede the time constant word to indicate that the next word is a time constant. the time constant word can be any value from 1 to 256 (Figure 6). Note that 00_{16} is interpreted as 256.

In timer mode, the time interval is controlled by three factors:

- The system clock period (ϕ)
- The prescaler factor (P), which multiplies the interval by either 16 or 256
- The time constant (T), which is programmed into the time constant register.

Consequently, the time interval is the product of $\phi \times P \times T$. The minimum timer resolution is $16 \times \phi$ (4 μs with a 4 MHz clock). The maximum timer interval is $256 \times \phi \times 256$ (16.4 ms with a 4 MHz clock). For longer intervals timers may be cascaded.

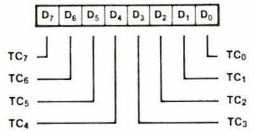


Figure 6. Time Constant Word

Programming (Continued)

Interrupt Vector Programming. If the CTC has one or more interrupts enabled, it can supply interrupt vectors to the CPU. To do so, the CTC must be pre-programmed with the most-significant five bits of the interrupt vector. Programming consists of writing a vector word to the I/O port corresponding to the CTC Channel 0. Note that D_0 of the vector word is always zero, to distinguish the vector from a channel control word. D_1 and D_2 are not used in programming the vector word. These bits are supplied by the interrupt logic to identify the channel requesting interrupt service with a unique interrupt vector (Figure 7). Channel 0 has the highest priority.

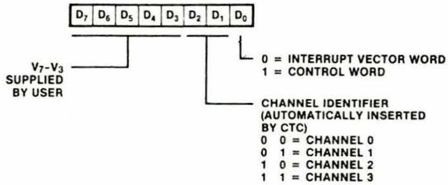


Figure 7. Interrupt Vector Word

Pin Description

$\overline{CE}$. *Chip Enable* (input, active Low). When enabled the CTC accepts control words, interrupt vectors, or time constant data words from the data bus during an I/O write cycle; or transmits the contents of the down-counter to the CPU during an I/O read cycle. In most applications this signal is decoded from the eight least significant bits of the address bus for any of the four I/O port addresses that are mapped to the four counter-timer channels.

CLK. *System Clock* (input). Standard singlephase Z80C system clock.

CLK/TRG$_0$-CLK/TRG$_3$. *External Clock/Timer Trigger* (input, user-selectable active High or Low). Four pins corresponding to the four CTC channels. In counter mode, every active edge on this pin decrements the down-counter. In timer mode, an active edge starts the timer.

CS$_0$-CS$_1$. *Channel Select* (inputs active High). Two-bit binary address code selects one of the four CTC channels for an I/O write or read (usually connected to A_0 and A_1).

D$_0$-D$_7$. *System Data Bus* (bidirectional, 3-state). Transfers all data and commands between the CPU and the CTC.

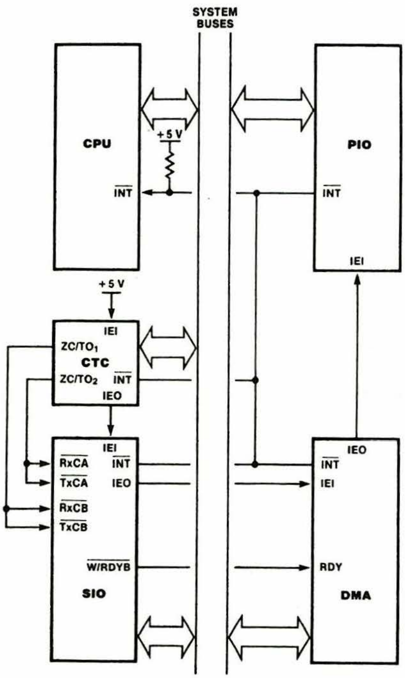


Figure 8. A Typical Z80C Environment

Pin Description (Continued)

IEI. *Interrupt Enable In* (input, active High). A High indicates that no other interrupting devices of higher priority in the daisy chain are being serviced by the CPU.

IEO. *Interrupt Enable Out* (output, active High). High only if IEI is High and the CPU is not servicing an interrupt from any CTC channel. IEO blocks lower priority devices from interrupting while a higher priority interrupting device is being serviced.

$\overline{\text{INT}}$. *Interrupt Reqeust* (output, open drain, active Low). Low when any CTC channel that has been programmed to enable interrupts has a zero-count condition in its down-counter.

$\overline{\text{IORQ}}$. *Input/Output Request* (input from CPU, active Low). Used with $\overline{\text{CE}}$ and $\overline{\text{RD}}$ to transfer data and channel control words between the CPU and the CTC. During a write cycle $\overline{\text{IORQ}}$ and $\overline{\text{CE}}$ are active and $\overline{\text{RD}}$ inactive. The CTC does not receive a specific write signal; rather, it internally generates its own from the inverse of an active $\overline{\text{RD}}$ signal. In a read cycle, $\overline{\text{IORQ}}$, $\overline{\text{CE}}$ and $\overline{\text{RD}}$ are active; the contents of the of the down-counter are read by the CPU. If $\overline{\text{IORQ}}$ and $\overline{\text{M1}}$ are both true, the CPU is acknowledging an interrupt request, and the highest priority interrupting channel places its interrupt vector on the data bus.

$\overline{\text{M1}}$. *Machine Cycle One* (input from CPU, active Low). When $\overline{\text{M1}}$ and $\overline{\text{IORQ}}$ are active, the CPU is acknowledging an interrupt. The CTC then places an interrupt vector on the data bus if it has highest priority, and if a channel has requested an interrupt ($\overline{\text{INT}}$).

$\overline{\text{RD}}$. *Read Cycle Status* (input, active Low). Used in conjunction with $\overline{\text{IORQ}}$ and $\overline{\text{CE}}$ to transfer data and channel control words between the CPU and the CTC.

$\overline{\text{RESET}}$. *Reset* (input active Low). Terminates all down-counts and disables all interrupts by resetting the interrupt bits in all control registers; the ZC/TO and the Interrupt outputs go inactive; IEO reflects IEI; D_0-D_7 go to the high-impedance state.

ZC/TO_0-ZC/TO_2. *Zero Count/Timeout* (output, active High). Three ZC/TO pins corresponding to CTC channels 2 through 0 (Channel 3 has no ZC/TO pin). In both counter and timer modes the output is an active High pulse when the down-counter decrements to zero.

Timing

Read Cycle Timing. Figure 9 shows read cycle timing. This cycle reads the contents of a down-counter without disturbing the count. During clock cycle T_2, the CPU initiates a read cycle by driving the following inputs Low: $\overline{\text{RD}}$, $\overline{\text{IORQ}}$, and $\overline{\text{CE}}$. A 2-bit binary code at inputs CS_1 and CS_0 selects the channel to be read. $\overline{\text{M1}}$ must be High to distinguish this cycle from an interrupt acknowledge. No additional wait states are allowed.

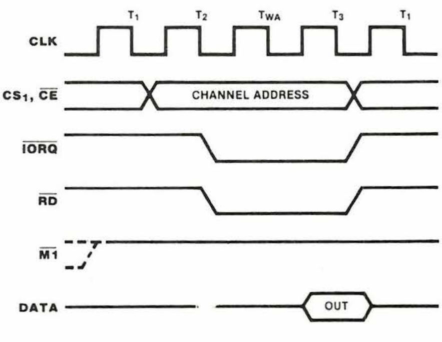


Figure 9. Read Cycle Timing

Timing (Continued)

Write Cycle Timing. Figure 10 shows write cycle timing for loading control, time constant or vector words.

The CTC does not have a write signal input, so it generates one internally when the read ($\overline{RD}$) input is High during T_1. During T_2 $\overline{IORQ}$ and $\overline{CE}$ inputs are Low. $\overline{M1}$ must be High to distinguish a write cycle from an interrupt acknowledge. A 2-bit binary code at inputs CS_1 and CS_0 selects the channel to be addressed, and the word being written is placed on the data bus. The data word is latched into the appropriate register with the rising edge of clock cycle T_3.

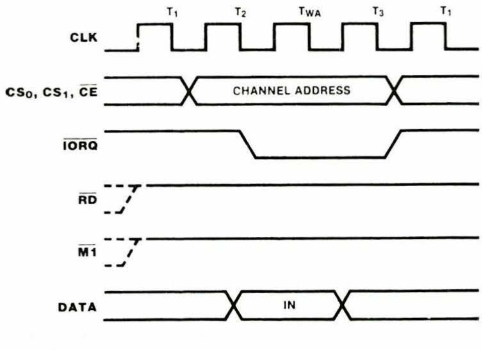


Figure 10. Write Cycle Timing

Timer Operation. In the timer mode, a CLK/TRG pulse input starts the timer (Figure 11) on the second succeeding rising edge of CLK. The trigger pulse is asynchronous and it must have a minimum width. A minimum lead time (210 ns) is required between the active edge of the CLK/TRG and the next rising edge of CLK to enable the prescaler on the following clock edge. If the CLK/TRG edge occurs closer than this, the initiation of the timer function is delayed one clock cycle. This corresponds to the startup timing discussed in the programming section. The timer can also be started automatically if so programmed by the channel control word.

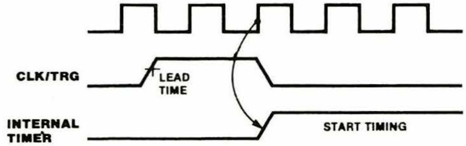


Figure 11. Timer Mode Timing

Counter Operation. In the counter mode, the CLK/TRG pulse input decrements the downcounter. The trigger is asynchronous, but the count is synchronized with CLK. For the decrement to occur on the next rising edge of CLK, the trigger edge must precede CLK by a minimum lead time as shown in Figure 12. If the lead time is less than specified, the count is delayed by one clock cycle. The trigger pulse must have a minimum width, and the trigger period must be at least twice the clock period.

The ZC/TO output occurs immediately after zero count, and follows the rising CLK edge.

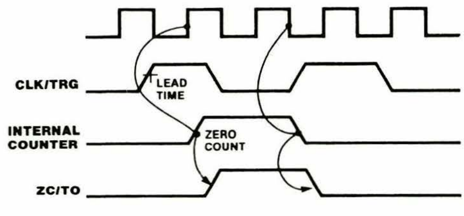


Figure 12. Counter Mode Timing

Interrupt Operation

The CTC follows the Z80C system interrupt protocol for nested priority interrupts and return from interrupt, wherein the interrupt priority of a peripheral is determined by its location in a daisy chain. Two lines—IEI and IEO—in the CTC connect it to the system daisy chain. The device closest to the +5 V supply has the highest priority (Figure 13). For additional information on the Z80C interrupt structure, refer to the *Z80 CPU Technical Manual.*

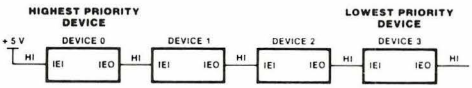


Figure 13. Daisy-Chain Interrupt Priorities

Within the CTC, interrupt priority is predetermined by channel number: Channel 0 has the highest priority, and Channel 3 the lowest. If a device or channel is being serviced with an interrupt routine, it cannot be interrupted by a device or channel with lower priority until service is complete. Higher priority devices or channels may interrupt the servicing of lower priority devices or channels.

A CTC channel may be programmed to request an interrupt every time its down-counter reaches zero. Note that the CPU must be programmed for interrupt mode 2. Some time after the interrupt request, the CPU sends an interrupt acknowledge. The CTC interrupt control logic determines the highest priority channel that is requesting an interrupt. Then, if the CTC IEI input is High (indicating that it has priority within the system daisy chain) it places an 8-bit interrupt vector on the system data bus. The high-order five bits of this vector were written to the CTC during the programming process; the next two bits are provided by the CTC interrupt control logic as a binary code that identifies the highest priority channel requesting an interrupt; the low-order bit is always zero.

Interrupt Acknowledge Timing. Figure 14 shows interrupt acknowledge timing. After an interrupt request, the CPU sends an interrupt acknowledge ($\overline{M1}$ and $\overline{IORQ}$). All channels are inhibited from changing their interrupt request status when $\overline{M1}$ is active—about two clock cycles earlier than $\overline{IORQ}$. $\overline{RD}$ is High to distinguish this cycle from an instruction fetch.

The CTC interrupt logic determines the highest priority channel requesting an interrupt. If the CTC interrupt enable input (IEI) is High, the highest priority interrupting channel within the CTC places its interrupt vector on the data bus when $\overline{IORQ}$ goes Low. Two wait states (T_{WA}) are automatically inserted at this time to allow the daisy chain to stabilize. Additional wait states may be added.

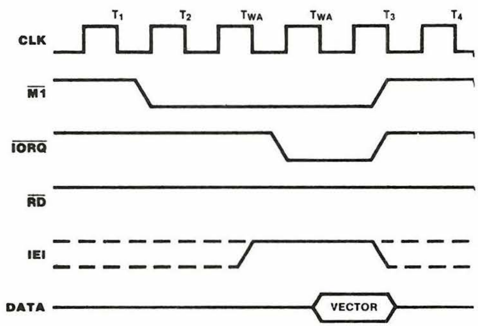


Figure 14. Interrupt Acknowledge Timing

Interrupt Operation (Continued)

Return from Interrupt Timing. At the end of an interrupt service routine the RETI (Return From Interrupt) instruction initializes the daisy chain enable lines for proper control of nested priority interrupt handling. The CTC decodes the 2-byte RETI code internally and determines whether it is intended for a channel being serviced. Figure 15 shows RETI timing.

If several Z80C peripherals are in the daisy chain, IEI settles active (High) on the chip currently being serviced when the opcode ED_{16} is decoded. If the following opcode is $4D_{16}$, the peripheral being serviced is released and its IEO becomes active. Additional wait states are allowed.

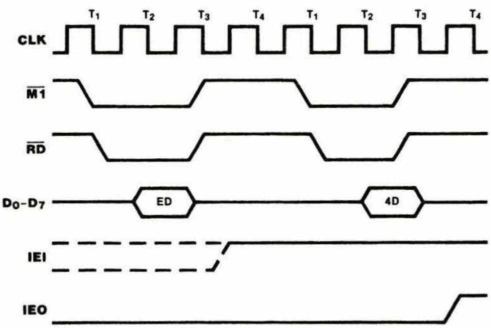
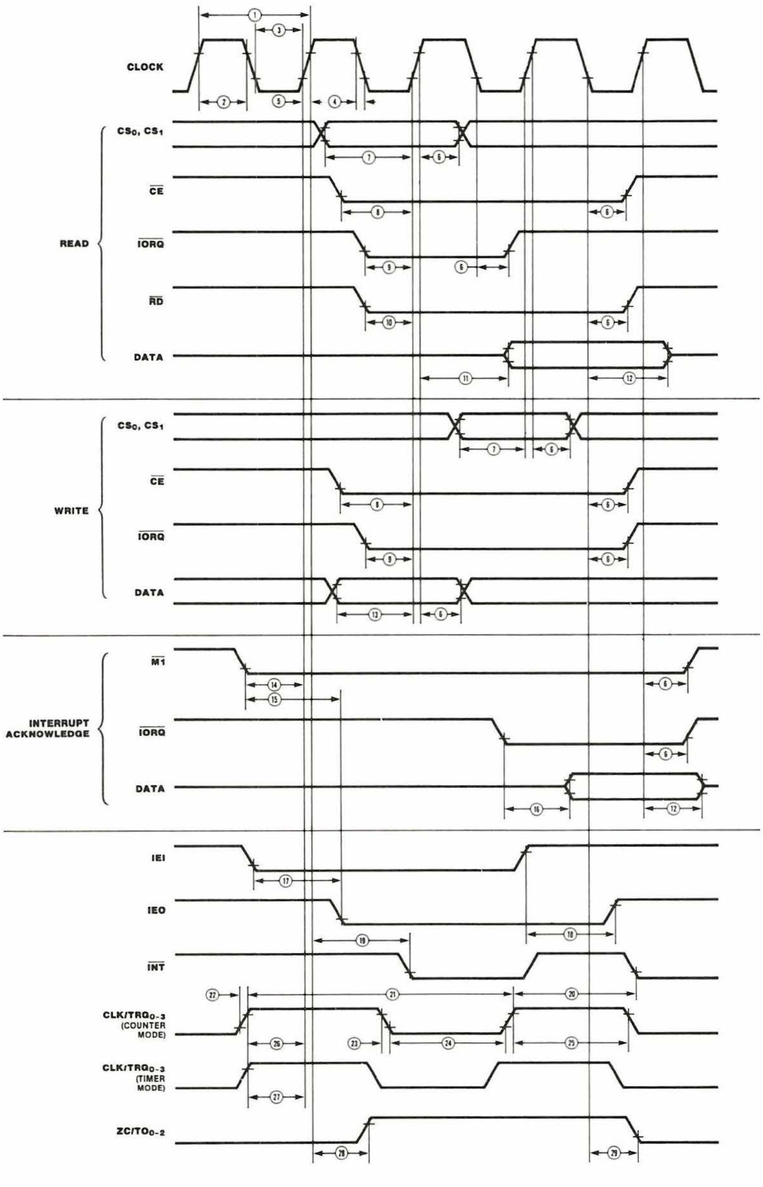


Figure 15. Return From Interrupt Timing

AC Characteristics



AC Characteristics (Continued)

Number	Symbol	Parameter	Z84C30A Min (ns)	Z84C30A Max (ns)
1	TcC	Clock Cycle Time	250	—
2	TwCh	Clock Width (High)	105	—
3	TwCl	Clock Width (Low)	105	—
4	TfC	Clock Fall Time	—	30
5	TrC	Clock Rise Time	—	30
6	Th	All Hold Times	0	—
7	TsCS(C)	CS to Clock ↑ Setup Time	160	—
8	TsCE(C)	$\overline{CE}$, to Clock ↑ Setup Time	150	—
9	TsIO(C)	$\overline{IORQ}$ ↓ to Clock ↑ Setup Time	115	—
10	TsRD(C)	$\overline{RD}$, ↓ to Clock ↑ Setup Time	115	—
11	TdC(DO)	Clock ↑ to Data Out Delay	—	200
12	TdC(DOz)	Clock ↓ to Data Out Float Delay	—	110
13	TsDI(C)	Data In to Clock ↑ Setup Time	50	—
14	TsMI(C)	$\overline{MI}$ to Clock ↑ Setup Time	90	—
15	TdMI(IEO)	$\overline{MI}$ ↓ to IEO ↓ Delay (Interrupt Immediately Preceding $\overline{MI}$)	—	190
16	TdIO(DOI)	$\overline{IORQ}$ ↓ to Data Out Delay (INTA Cycle)	—	160
17	TdIEI(IEOf)	IEI ↓ to IEO ↓ Delay	—	130
18	TdIEI(IEOr)	IEI ↑ to IEO ↑ Delay (After ED Decode)	—	160
19	TdC(INT)	Clock ↑ to $\overline{INT}$ ↓ Delay	—	(1)TcC + 140
20	TdCLK(INT)	CLK/TRG ↑ to $\overline{INT}$ ↓	—	(2)
		tsCTR(C) satisfied	—	TcC + 160
		tsCTR(C) not satisfied	—	2 TcC + 370
21	TcCTR	CLK/TRG Cycle Time	(2) 2TcC	
22	TrCTR	CLK/TRG Rise Time	—	50
23	TfCTR	CLK/TRG Fall Time	—	50
24	TwCTRl	CLK/TRG Width (Low)	200	—
25	TwCTRh	CLK/TRG Width (High)	200	—
26	TsCTR(Cs)	CLK/TRG ↑ to Clock ↑ Setup Time for Immediate Count	(2) 210	—
27	TsCTR(Cs)	CLK/TRG ↑ to Clock ↑ Setup Time for enabling of Prescaler on following clock ↑	(1) 210	—
28	TdC(ZC/TOr)	Clock ↑ to ZC/TO ↑ Delay	—	190
29	TdC(ZC/TOf)	Clock ↓ to ZC/TO ↓ Delay	—	190

Notes: (1) Timer mode
(2) Counter mode

Absolute Maximum Ratings

Symbol	Item	Rating
V_{CC}	V_{CC} Supply Voltage with respect to V_{SS}	$-0.5V$ to $7V$
V_{IN}	Input Voltage	$-0.5V$ to $V_{CC}+0.5V$
P_D	Power Dissipation (Ta = 85°C)	250mW
T_{SOLDER}	Soldering Temperature (Soldering Time 10 sec.)	260°C
T_{STG}	Storage Temperature	−65°C to 150°C
T_{OPR}	Operating Temperature	−40°C to 85°C

DC Characteristics (1)

Symbol	Parameter	Min.	Typ.	Max.	Unit	Test Conditions
V_{ILC}	Clock Input Low Voltage	−0.3	—	0.6	V	
V_{IHC}	Clock Input High Voltage	$V_{CC}-0.6$	—	$V_{CC}+0.3$	V	
V_{IL}	Input Low Voltage (except CLK)	−0.5	—	0.8	V	
V_{IH}	Input High Voltage (except CLK)	2.2	—	V_{CC}	V	
V_{OL}	Output Low Voltage	—	—	0.4	V	$I_{OL}=2.0mA$
V_{OH1}	Output High Voltage (1)	2.4	—	—	V	$I_{OH}=-1.6mA$
V_{OH2}	Output High Voltage (2)	$V_{CC}-0.8$	—	—	V	$I_{OH}=-250\mu A$
I_{LI}	Input Leakage Current	—	—	±10	μA	$V_{SS} \leq V_{IN} \leq V_{CC}$
I_{LO}	3-State Output Leakage Current in Float	—	—	±10	μA	$V_{SS}+0.4 \leq V_{OUT} \leq V_{CC}$
I_{CC1}	Operating Supply Current	—	3	7	mA	$V_{CC}=5V$, CLK = 4MHz $V_{IH}=V_{CC}-0.2V$ $V_{IL}=0.2V$
I_{CC2}	Stand-by Supply Current	—	0.5	10	μA	$V_{CC}=5V$ $CLK=V_{CC}$ $V_{IH}=V_{CC}-0.2V$ $V_{IL}=0.2V$
I_{OHD}	Darlington Drive Current (1)	−1.5	—	−5.0	mA	$V_{OH}=1.5V$, $R_{EXT}=1.1k\Omega$

Note (1): Applied to ZC/TO_0, ZC/TO_1 and ZC/TO_2.

Test Conditions

T_A = −40°C To +85°C
V_{CC} = 5 V ± 10%
V_{SS} = 0 V.

AC test Conditions

- Inputs except CLK (clock) are driven at 2.4 V for a logic "1" and 0.4 V for a logic "0". Clock input is driven at $V_{CC}-0.6$ V for a logic "1" and 0.6 V for a logic "0".
- Timing measurements are made at 2.2 V for a logic "1" and 0.8 V for a logic "0".

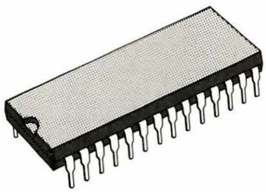
All AC parameters assume a load capacitance of 100 pF.

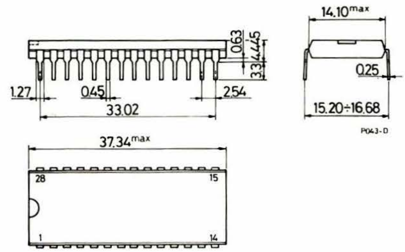
Ordering Information

Type	Package	Temp.	Clock	Description
Z84C30A B6	Plastic	−40/+85°C	4 MHz	Z80C Counter Timer Control
Z84C30A F6	Frit Seal	−40/+85°C		CMOS Version
Z84C30A C6	Plastic Chip Carrier	−40/+85°C		
Z84C30A K6	Ceramic Chip Carrier	−40/+85°C		

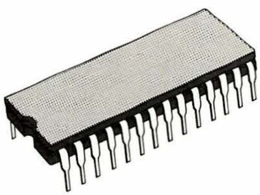
Packages (dimension in mm)

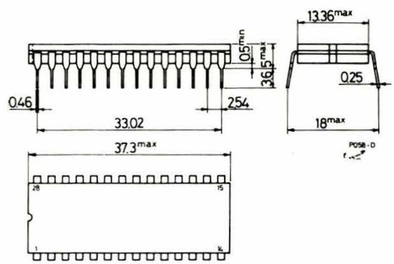
28 - Lead Plastic DIP





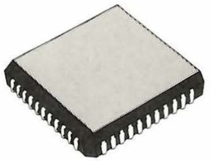
28 - Lead Ceramic DIP (Frit-Seal)

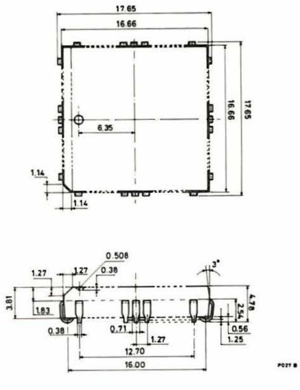




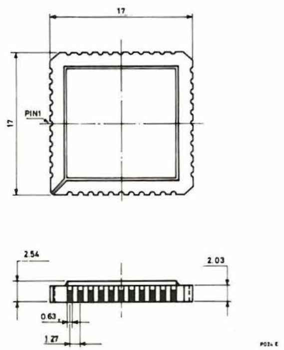
Packages (Continued)

44 - Leaded Plastic Chip Carrier





44 - Leadless Ceramic Chip Carrier



SGS GROUP OF COMPANIES

Italy - Brazil - France - Malta - Malaysia - Singapore - Sweden - Switzerland - United Kingdom - U.S.A. - West Germany

SGS COMMERCIAL NETWORK

INTERNATIONAL HEADQUARTERS
SGS Microelettronica SpA
Via C. Olivetti 2, - 20041 Agrate Brianza-Italy
Tel.: 039 - 65551
Telex: 330131 - 330141 - SGSAGR

BENELUX
SGS Microelettronica SpA
Sales Office:
Bruxelles 1040
Bld. Reyerslaan, 207 209
Tel.: 02 - 7366060
Telex: 24149

BRAZIL
SGS Semicondutores LTDA
Sales Office:
05413 Sao Paulo
Av. Henrique Schaumann 286 - CJ33
Tel.: 011 - 853-5062
Telex: 37988 UMBR BR

DENMARK
SGS Semiconductor A.B.
Sales Office:
2730 Herlev
Herlev Torv, 4
Tel.: 02 - 948533
Telex: 35411

FRANCE
Société Générale de Semiconducteurs
92120 Montrouge
21-23 Rue de la Vanne
Tel.: 01 - 6571133
Telex: 250938F

HONG KONG
SGS Semiconductor Asia Limited
Hunghom, Kowloon
9th Floor, Block N,
Kaiser Estate, Phase III,
11 Hok Yuen St.,
Tel.: 03-644251/5
Telex: 63906 ESGIE HX

ITALY
SGS Microelettronica SpA
Direzione Italia e Sud Europa
20090 Assago (MI)
V.le Milanofiori - Strada 4 - Palazzo A/4/A
Tel.: 02 - 8244131 (10 linee)
Telex: 330131 - 330141 SGSAGR

Sales Offices:
40128 Bologna
Via Corticella, 231
Tel.: 051-324486
00161 Roma
Via A. Torlonia, 15
Tel.: 06-8444474

KOREA
SGS Semiconductor (Pte) Ltd.
Korea Liason Office
Mapo, Seoul 121
Rm 1306 KMIC Bldg
168-9 Yumlidong
Tel.: 712-7071/2/3
Telex: K 26493

SINGAPORE
SGS Semiconductor (Pte) Ltd.
Singapore 2056
28 Ang Mo Kio
Industrial Park 2
Tel.: 482-1411
Telex: RS 55201 ESGIES

SWEDEN
SGS Semiconductor A.B.
19500 Märsta
Bristagatan, 16
Tel.: 0760 - 40120
Telex: 042 - 10932

SWITZERLAND
SGS Semiconductor S.A.
Sales Offices:
1218 Grand-Saconnex (Geneve)
Chemin François-Lehmann, 22
Tel.: 022 - 986462/3
Telex: 28895
8048 Zürich
Rautistrasse, 58
Tel. 01-4926606/41
Telex: 823573

TAIWAN-REPUBLIC OF CHINA
SGS Singapore (Pte) Ltd
Taipei Sec 4
6th floor, Pacific Commercial Bldg.
285 Chung Hsiao E Road
Tel.: 2-7728203
Telex: 10310 ESGIETWN

UNITED KINGDOM
SGS Semiconductor Limited
Aylesbury, Bucks
Planar House, Walton Street
Tel.: 0296 - 5977
Telex: 041 - 83245

WEST GERMANY
SGS Halbleiter Bauelemente GmbH
8018 Grafing bei München
Haidling, 17
Tel.: 08092-690
Telex: 05 27378

Sales Offices:
3012 Langenhagen
Hans Boeckler Str., 2
Tel.: 0511 - 789881
Telex: 923195
8500 Nürnberg 40
Allersberger Str., 95
eingang Wilhelminenstr. 1
Tel.: 0911 - 464071
Telex: 626243
8023 Pullach bei München
Seitnerstrasse, 42
Tel.: 089 - 793 0662
Telex: 5215784
7000 Stuttgart 31
Loewenmarkt, 5
Tel.: 0711 - 88110
Telex: 723625

U.S.A.
SGS-Semiconductor Corporation
Phoenix, AZ 85022
1000 East Bell Road
Tel.: (602) 867-6100
Telex: 249976 SGSPH UR

Sales Offices:
Austin, TX 78723
6448 Highway 290 East
Suite F-103
Tel.: (512) 459-4545
Bloomington, MN 55438
6100 Green Valley Drive
Suite 120
Tel.: (612) 835-1347
Telex: 201692 SGSMI UR
Ft. Lauderdale FL 33309
1001 NW 62nd Street
Suite 314
Tel.: (305) 4938881
Telex: 291588
Hauppauge, NY 11788
330 Motor Parkway
Suite 100
Tel.: (516) 435-1050
Telex: 221275 SGSHA UR
Indianapolis, IN 46241
2346 S. Lynhurst Drive
Suite H-200
Tel.: (317) 241-1116
Telex: 209144 SGSIN UR
Irvine, CA 92714
18271 W. McDurmott Drive
Suite J.
Tel. (714) 863-1222
Telex: 277793 SGSOR UR
Norcross, GA 30071
3040-D1 Holcomb Bridge Road
Tel.: (404) 446-8686
Telex: 261395 SGSAT UR
Plano, TX 75074
850 East Central Parkway
Suite 180
Tel.: (214) 881-0848
Telex: 203997 SGSDA UR
Poughkeepsie, NY 12601
201 South Avenue
Suite 206
Tel.: (914) 473-2255
Santa Clara, CA 95051
2700 Augustine Drive
Suite 209
Tel.: (408) 727-3404
Telex: 278833 SGSSA UR
Schaumburg, IL 60196
600 North Meacham Road
Tel.: (312) 490-1890
Telex: 210159 SGSCH UR
Waltham, MA 02154
240 Bear Hill Road
Tel.: (617) 890-6688
Telex: 200297 SGSWH UR
Westlake Village, CA 91361
31416 West Agoura Road
Suite 200
Tel.: (818) 991-2900
Telex: 215258 SGSWD UR
Woodland Hills, CA 91367
6355 Topanga Canyon Boulevard
Suite 220
Tel.: (213) 716-6600
Telex: 215258 SGSWD UR

ORDER CODE: DAZ80CMOSF/85